Anisotropic mesh generation

Mael Rouxel-Labbé

UNIVERSITÉ CÔTE D'AZUR

ED STIC

École Doctorale STIC
Sciences et Technologies de l'Information et de la Communication

Génération de Maillages Anisotropes

THÈSE

présentée et soutenue publiquement le 16 décembre 2016

pour l'obtention du

Doctorat de l'Université Côte d'Azur

(mention informatique)

par

Mael Rouxel-Labbé

Thèse dirigée par Jean-Daniel BOISSONNAT
co-encadrée par Jane TOURNOIS

Composition du jury

Président :	Pierre ALLIEZ	INRIA Sophia Antipolis – Méditerranée
Rapporteurs :	Mathieu DESBRUN	California Institute of Technology
	Jean-Marie MIREBEAU	Faculté des Sciences d'Orsay Université Paris-Sud
	Jonathan SHEWCHUK	University of California, Berkeley
Examinateurs :	Jean-Daniel BOISSONNAT	INRIA Sophia Antipolis – Méditerranée
	André LIEUTIER	Dassault Systèmes
	Jane TOURNOIS	GeometryFactory
Invitée :	Mariette YVINEC	INRIA Sophia Antipolis – Méditerranée (Emeritus)

Geometry Factory

informatics mathematics

Mis en page avec la classe thesul.

Acknowledgments

First and foremost, I would like to express my gratitude to my supervisor, Jean-Daniel Boissonnat for his insightful advices, his patience and his immutable optimistic outlook, which all greatly helped during this thesis. I would also like to thank Andreas Fabri for trusting me with this work and for his constant support.

I am very thankful to Mathijs Wintraecken for the many and fruitful discussions and for his dedication and thoroughness as a co-author. I also thank Jane Tournois and Mariette Yvinec for their availability and rigor.

I am grateful to Mathieu Desbrun, Jean-Marie Mirebeau and Jonathan Shewchuk for accepting to review this thesis and for their comments. Thanks to Mathijs, Jane, and Rémy for proofreading this manuscript.

I would like to thank Pierre Alliez and Mathieu Desbrun for the multiple discussions on anisotropy and for sharing their on-going works on the subject, Laurent Rineau for always being available and effective on programming-related issues, Laurent Busé for his help with bivariate polynomials, and Clément Jamin and Marc Glisse for technical discussions on the tangential complex.

I would also like to thank Frédéric Hecht for pointing out INRIA's Geometrica team as a possible direction after my master.

I have shared various offices across these years, all enjoyable. I would like to thank Rémy, Aymeric, Alba, Siargey, Sonali, and Manish for the times in and outside of the office.

My parents have always pushed me and have kept an ever-enduring faith in me throughout the years and for this I am infinitely grateful. This thesis is dedicated to them as well as to my brother, Tifenn.

Finally, I deeply thank Elena for her presence every day and for keeping me alive during the redaction of this manuscript.

À mes parents et mon frère

Résumé

Nous étudions dans cette thèse la génération de maillages anisotropes basée sur la triangulation de Delaunay et le diagramme de Voronoi.

Nous considérons tout d'abord les maillages anisotropes localement uniformes, développés par Boissonnat, Wormser et Yvinec. Bien que l'aspect théorique de cette approche soit connu, son utilité pratique n'a été que peu explorée. Une étude empirique exhaustive est présentée et révèle les avantages, mais aussi les inconvénients majeurs de cette méthode.

Dans un second temps, nous étudions les diagrammes de Voronoi anisotropes définis par Labelle et Shewchuk. Nous donnons des conditions suffisantes sur un ensemble de points pour que le dual du diagramme soit une triangulation plongée en toute dimension; un algorithme générant de tels ensembles est conçu. Ce diagramme est utilisé pour concevoir un algorithme qui génère efficacement un maillage anisotrope pour des domaines de dimension intrinsèque faible plongés dans des espaces de dimension large. Notre algorithme est prouvable, mais les résultats sont décevants.

Enfin, nous présentons le diagramme de Voronoi Riemannien discret, qui utilise des avancées récentes dans l'estimation de distances géodésiques et dont le calcul est grandement accéléré par l'utilisation d'un graphe anisotrope. Nous donnons des conditions suffisantes pour que notre structure soit combinatoirement équivalente au diagramme de Voronoi Riemannien et que son dual utilisant des simplexes droits mais aussi courbes est une triangulation plongée en toute dimension. Nous obtenons de bien meilleurs résultats que pour nos autres techniques, mais dont l'utilité reste limitée.

Mots-clés: Anisotropie, Génération de maillages, Triangulation de Delaunay, Diagramme de Voronoi, Géometrie Riemannienne.

Abstract

In this thesis, we study the generation of anisotropic meshes using the concepts of Delaunay triangulations and Voronoi diagrams.

We first consider the framework of locally uniform anisotropic meshes introduced by Boissonnat, Wormser and Yvinec. Despite known theoretical guarantees, the practicality of this approach has only been hardly studied. An exhaustive empirical study is presented and reveals the strengths but also the overall impracticality of the method.

In a second part, we investigate the anisotropic Voronoi diagram introduced by Labelle and Shewchuk and give conditions on a set of seeds such that the corresponding diagram has a dual that is an embedded triangulation in any dimension; an algorithm to generate such sets is devised. Using the same diagram, we propose an algorithm to generate efficiently anisotropic triangulations of low-dimensional manifolds embedded in high-dimensional spaces. Our algorithm is provable, but produces disappointing results.

Finally, we study Riemannian Voronoi diagrams and introduce discrete Riemannian Voronoi diagrams, which employ recent developments in the numerical computation of geodesic distances and whose computation is accelerated through the use of an underlying anisotropic graph structure. We give conditions that guarantee that our discrete structure is combinatorially equivalent to the Riemannian Voronoi diagram and that its dual is an embedded triangulation, using both straight and curved simplices. We obtain significantly better results than with our other methods, but the overall utility of our techniques remains limited.

Keywords: Anisotropy, Mesh generation, Delaunay triangulation, Voronoi diagram, Riemannian Geometry.

Contents

List of Symbols

Domains

$\mathcal{P}$	Point set
Ω	Domain
n	Ambient dimension
$\mathcal{M}$	Smooth manifold
m	Dimension of $\mathcal{M}$
$T_p\mathcal{M}$	Tangent space of $\mathcal{M}$ at p
$\mathcal{Q}$	Fixed paraboloid
$\mathcal{C}$	Canvas
$h_{\mathcal{C}}$	Sizing field of the canvas

Metric

G	Metric
F	Square root of a metric
λ_i, v_i	Eigenvalues and eigenvectors of a metric
$\psi(G_1, G_2)$	Distortion between two metrics G_1 and G_2
$g_{\mathbb{E}}$	Euclidean metric field
g_0	Uniform metric field
g	Arbitrary metric field
ζ	Lipschitz coefficient (of a Lipschitz-continuous metric field)
d_G	Distance with respect to the metric G
$\|\cdot\|_G$	Norm with respect to the metric G
$d_{\mathbb{E}}$	Distance with respect to the Euclidean metric $\mathbb{E}$
d_g	Geodesic distance with respect to the metric field g
$S_g(x, r)$	Geodesic sphere centered on x and of radius r
$B_g(x, r)$	Geodesic ball centered on x and of radius r

Voronoi and Delaunay

$\mathcal{N}(C)$	Nerve of a covering C
$\mathcal{K}$	Simplicial complex
$\mathrm{Vor}_d(\mathcal{P})$	Voronoi diagram of $\mathcal{P}$ using the distance d
$\mathrm{Vor}_g(\mathcal{P})$	Riemannian Voronoi diagram of $\mathcal{P}$ with respect to g
$\mathrm{Vor}_g^d(\mathcal{P})$	Discrete Riemannian Voronoi diagram of $\mathcal{P}$ with respect to g
$\mathrm{Del}(\mathcal{P})$	Abstract Delaunay complex of the point set $\mathcal{P}$
$\mathrm{Del}_G(\mathcal{P})$	Delaunay complex of the point set $\mathcal{P}$ with respect to G
$\mathrm{Del}_{g_0}(\mathcal{P})$	Delaunay complex of the point set $\mathcal{P}$ with respect to g_0 (uniform metric field)
$\mathrm{Del}^{\omega}(\mathcal{P})$	Regular Delaunay complex of the point set $\mathcal{P}$ with weights ω
$\widetilde{\mathrm{Del}}_g(\mathcal{P})$	Curved Riemannian Delaunay complex of the point set $\mathcal{P}$ with respect to g
$\overline{\mathrm{Del}}_g(\mathcal{P})$	Straight Riemannian Delaunay complex of the point set $\mathcal{P}$ with respect to g
$\mathcal{B}_g(\sigma)$	Delaunay ball of the simplex σ in the metric g
$\mathrm{V}_g(p)$	Riemannian Voronoi cell of the point $p \in \mathcal{P}$ in $\mathrm{Vor}_g(\mathcal{P})$
$\partial\,\mathrm{V}_g(p)$	Boundary of the Riemannian Voronoi cell $V_g(p)$
$\mathrm{V}_g^d(p)$	Discrete Voronoi cell of the point $p \in \mathcal{P}$ in $\mathrm{Vor}_g(\mathcal{P})$
$\mathrm{BS}_g(p,q)$	Bisector in $\mathrm{Vor}_g(\mathcal{P})$ of the seeds $p, q \in \mathcal{P}$
$\mathrm{BS}_g(\mathcal{P}')$	Bisector in $\mathrm{Vor}_g(\mathcal{P})$ of the seeds $p \in \mathcal{P}' \subset \mathcal{P}$
$\mathrm{V}_g^{+\omega}(p)$	Voronoi cell of the point $p \in \mathcal{P}$ in $\mathrm{Vor}_g(\mathcal{P})$ relaxed by a coefficient ω
$\mathrm{EV}_g(p)$	Eroded Voronoi cell of the point $p \in \mathcal{P}$ in $\mathrm{Vor}_g(\mathcal{P})$
$\mathrm{DV}_g(p)$	Dilated Voronoi cell of the point $p \in \mathcal{P}$ in $\mathrm{Vor}_g(\mathcal{P})$
σ, τ	Simplices
$D(p,\sigma)$	Altitude of the vertex p in the simplex σ

Sampling

ε	Sampling criterion of a point set
μ	Separation criterion of a point set
λ	Separation coefficient: $\mu = \lambda\varepsilon$
δ	Power protection criterion of a point set
ι	Power protection coefficient: $\delta = \iota\varepsilon$

ε_0	Sampling criterion of a point set with respect to a uniform metric field
μ_0	Separation criterion of a point set with respect to a uniform metric field
δ_0	Power protection criterion of a point set with respect to a uniform metric field

Locally uniform anisotropic meshes

ψ_0	Distortion criterion parameter
r_0	Sizing criterion parameter
ρ_0	Shape criterion parameter
σ_0	Sliverity parameter
β	Acceptable inconsistency scale parameter
δ	Pick valid zone size parameter
S_p	Star of p
S_p^v	Restricted volume star of p
S_p^s	Restricted surface star of p
$\mathrm{PV}(\tau)$	Picking zone of the simplex τ
Z_p	Conflict zone of S_p

Riemannian Geometry

K	Sectional curvature
Λ_-, Λ_+	Bounds on the sectional curvature
$\iota_{\mathcal{M}}$	Injectivity radius of $\mathcal{M}$
$\mathcal{H}$	Hyperplane
$d(p, \mathcal{H})$	Smallest distance between p and $q \in \mathcal{H}$

General Introduction

To visualize, simulate and understand phenomena and systems of our world, models are developed. As an exact description is often unknown or undesirable – because too complicated – these models generally seek to offer a simplified yet faithful representation of these phenomena and systems. Scientific models are for example used to represent the currents in the atmosphere and the oceans in climatology, predict the trajectories of astronomical objects, or understand earthquakes and determine the composition of planets in seismology. However, modeling is not limited to the study of natural phenomena: geometric modeling, which describes shapes, finds uses in the motion planning of robots, the design of buildings in architecture and machines in mechanical engineering, and in many more domains such as animation, topography, cartography...

A scientific model, while already being an approximation of a real world phenomenon, is often still too complex to be solved analytically. Numerical analysis, one of the oldest fields of mathematics, seeks to provide approximate yet accurate solutions to mathematical problems, ranging from the evaluation of square roots to the resolution of difficult partial differential equations. The use of computers to perform numerical analysis computations spawned the field of numerical simulation, whose capabilities and fields of application are ever expanding, thanks to the constant increase in the computational power of computers. However, the objects and domains of our continuous world – whether they are used for scientific or geometric modeling – cannot usually be exactly represented in the inherently finite systems of computers. Numerical simulation of systems and visualization of objects must thus first go through a discretization of the domains of interest. This is generally achieved by retaining only pointwise information from the real input, and linking these points together to form a network that approximates the domain – a mesh. Numerical techniques and algorithms, such as spectral method, combine a scientific model and a mesh to approximately simulate a system and have imposed themselves as powerful and necessary tools in many different industries.

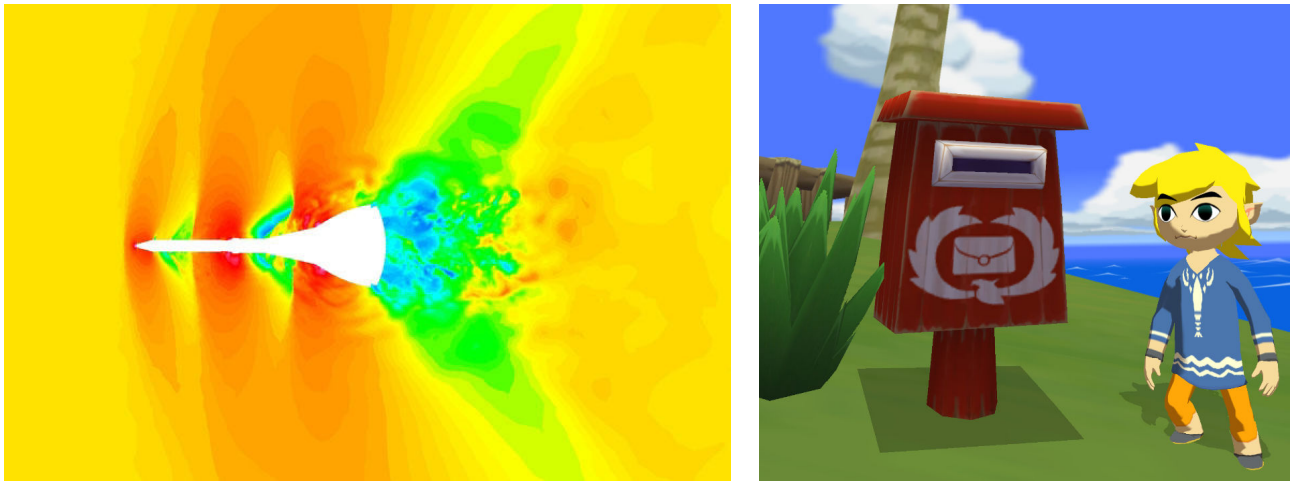

Figure 1: Two applications of meshes: the simulation of the pressure around the Orion capsule and its launch escape system (left, [1]) and a video game (right, [109]).

Despite a large number of theoretical and practical studies, the automatic generation of a mesh is still a challenging problem. Indeed, the large amount of applications creates a wide range of different types of domains that must be meshed, from intricate mechanical pieces to vast layers of crust, making it difficult to provide a single satisfactory algorithm. Additionally, the desired mesh of a given domain generally differs depending on the purpose of the mesh: in the context of simulation, the elements forming a mesh must for example allow the adaptation of the model to this discretization. Finally, the geometrical faithfulness of the mesh to the input and the shape of its elements are of tremendous importance to obtain a faithful simulation or representation. These requirements on shape of the elements of the mesh are diverse (angles, edge lengths, ...) and can be very specific to a model. For instance, to ensure the numerical stability of a simulation using the Finite Element method in a three-dimensional setting, it is necessary that the tetrahedra have bounded aspect ratio. The combination of these independent constraints have made mesh generation the typical bottleneck in modeling, pushing many industries to still spend weeks – and even months – generating meshes that fits their need manually.

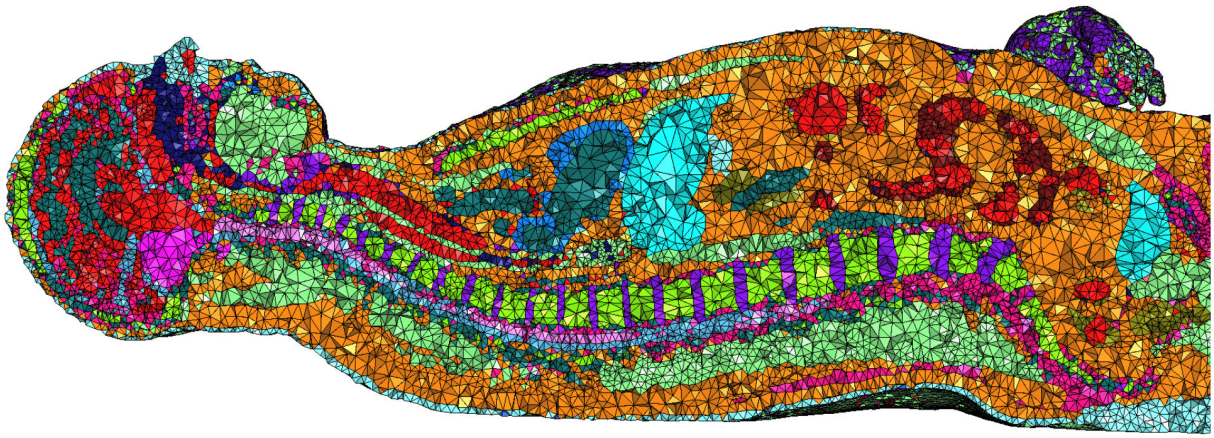

Figure 2: Multi-domain isotropic mesh of a human, [118].

Simplicial meshes

Meshes are generally made up of simple geometric elements for practical reasons: a mesh built using simple polygons (or polyhedra) is simpler to generate, implement, or modify; in the context of simulation, continuous models also are more easily adapted. For similar reasons, while it is possible to use so-called hybrid meshes that simultaneously combine different kinds of geometric elements, it is preferred to keep the same type of elements over the whole mesh. In the large amount of literature on the subject, various types of elements have been studied such as quads, cubes, pentagons, hexahedra.

Meshes can be divided into two categories: structured meshes, that use regularly placed points and identical elements, and unstructured meshes, whose points are irregularly placed and allow the shape of the elements to vary. Unstructured meshes have become very popular for their flexibility, allowing precise approximations of the domain. However, the increased faithfulness of the discretization is only obtained at the cost of a more difficult mesh generation process.

Simplicial meshes, whose elements are triangles in a two-dimensional setting, tetrahedra in a three-dimensional setting, and more generally n-simplices (the convex hull of $n+1$ points) in a n-dimensional setting, constitute one of the most common types of meshes due to the simplicity of the base element.

They have subsequently received much attention and are used in major numerical methods, such as for example the finite element method. The quality of a simplex, generally determined by its closeness to the regular simplex, is primordial to obtain accurate simulations and faithful representations. The generation of an unstructured simplicial mesh must therefore avoid the creation of poorly shaped simplices.

Delaunay triangulation

The Delaunay triangulation is an extensively studied simplicial mesh that is characterized by many useful properties. While different definitions can be given, the Delaunay triangulation of a set of points $\mathcal{P}$ is generally defined through the empty sphere property: the circumsphere of any simplex in a Delaunay triangulation is empty of any other vertex of $\mathcal{P}$. The Delaunay triangulation is characterized by many useful properties and can be shown to be unique under mild assumptions. Notably, it maximizes the smallest angle out of all the triangulations of a point set in a two-dimensional setting, thus providing shape guarantees on the simplices of the mesh.

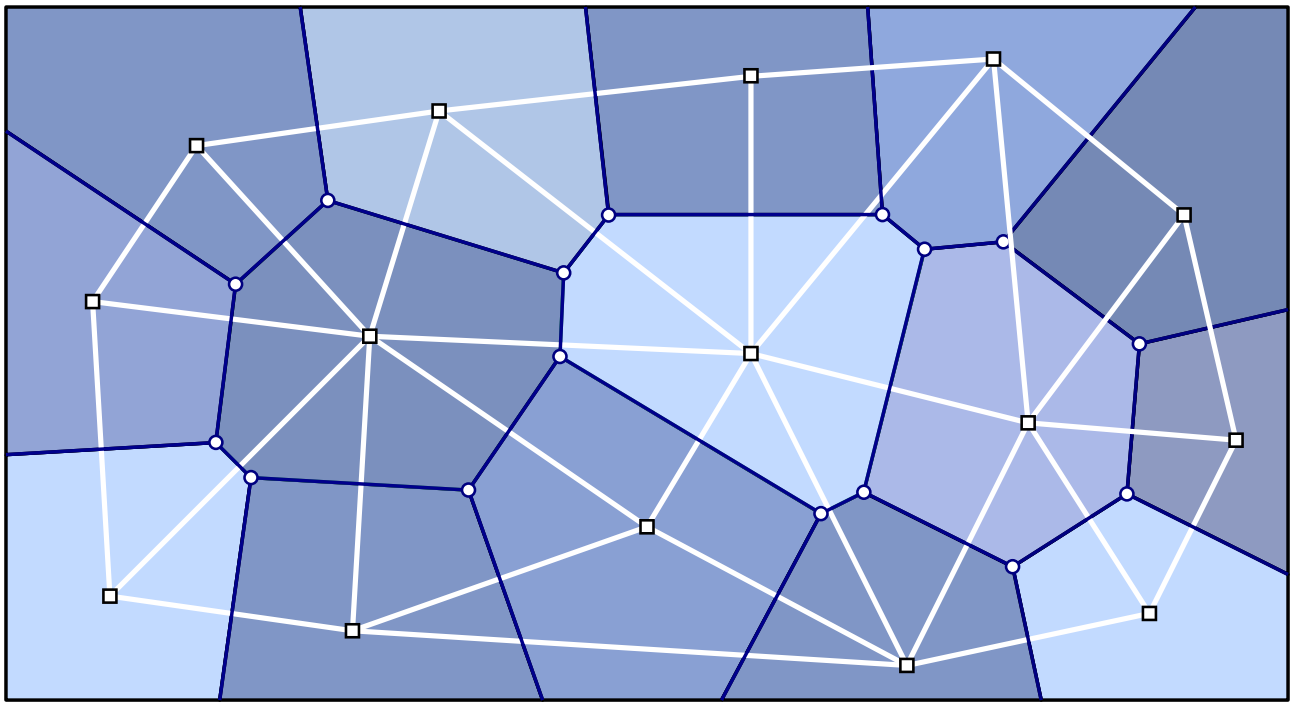

Figure 3: Delaunay triangulation (white) and Voronoi diagram (blue).

These good properties have made the Delaunay triangulation the basis of various structures, algorithms and methods in diverse fields, ranging from surface reconstruction to crystallography. Consequently, the generation of Delaunay meshes has received considerable attention and many ways are available to construct the Delaunay triangulation of a set of points $\mathcal{P}$. For example, edge flipping algorithms are able to reach the Delaunay triangulation starting from any other triangulation of $\mathcal{P}$. It can also be obtained by constructing the convex hull of the set of points $\mathcal{P}$ lifted to a higher-dimensional space. Finally, the Delaunay triangulation is intrinsically linked to another renowned structure of computational geometry, the Voronoi diagram. The Voronoi diagram of a set of points can be seen as a decomposition of space into domains of influence, also called Voronoi cells. The neighboring information of the cells of a Voronoi diagram of a set of points provides a natural graph structure between

points, which can be shown to be dual to the Delaunay triangulation of the points, thus providing another way to build the Delaunay triangulation.

As important as the connectivity of the mesh, the placement of the points is crucial to generate high-quality meshes. In early isotropic meshing techniques, Delaunay triangulations were constructed from existing point sets generated through other means. The seminal works of Ruppert and Chew incorporated the Delaunay triangulation in the generation of the point set: they introduced a two-dimensional refinement algorithm based on the iterative insertion of the circumcenter of simplices whose quality, in terms of size or circumradius to shortest edge ratio, is below a certain threshold. This refinement algorithm is proven to terminate and provides guarantees on the quality of the simplices. Building upon these results, more elaborate refinement algorithms have been devised to obtain robust and quality-guaranteeing mesh generation algorithms for complex domains such as surfaces with sharp features.

Anisotropic mesh generation

The ideal shape of simplex has so far been described as the regular simplex, but this is not always the case. Many physical systems and objects naturally exhibit anisotropy, that is the presence of privileged directions in the system: the flow of air, liquid or plasma in fluid dynamics and electromagnetism are examples of natural phenomena characterized by anisotropy.

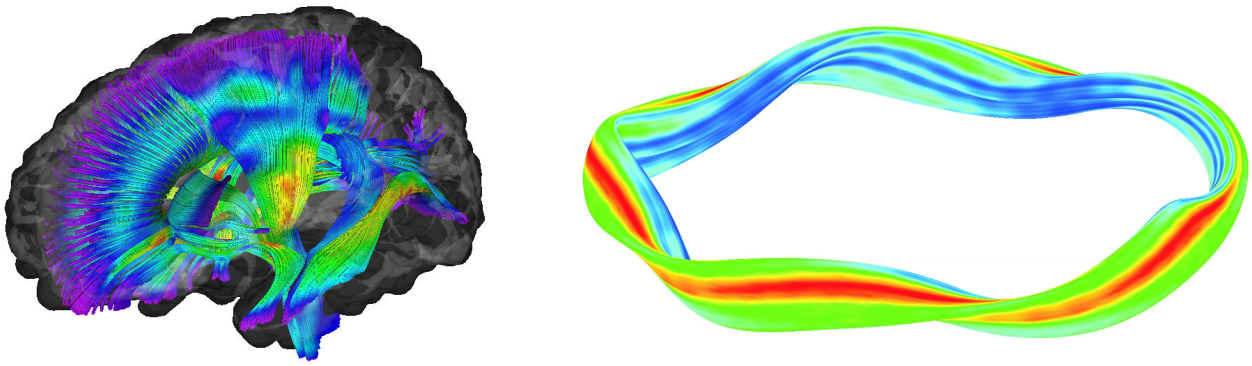

Figure 4: Simulation of fluids in a brain (left, [81]) and of a plasma chamber (right, [77]).

For models associated with anisotropy, the accuracy of the simulations has been shown to be enhanced when elements are adapted to the anisotropy, meaning that the simplices are elongated and follow the directionality of the anisotropy. From an equivalent point of view, the use of meshes whose elements are stretched according to the anisotropy of the phenomenon requires a lower number of elements to achieve the same precision of the result of the simulation. When stretched elements are used, the mesh is said to be anisotropic. Additionally to providing increased accuracy in the simulations of scientific modeling, anisotropic meshes also find use in geometric modeling as they can improve the visualization of objects, and lower the number of vertices required to represent a shape or interpolate a smooth function. By requiring fewer elements, anisotropic meshes thus provide another way to accelerate mesh generation, and increase the quality and the speed of computations. Where the main improvements in the accuracy of modeling only came from increased computation power and denser meshes for isotropic meshes, anisotropic mesh generation offers another independent way to obtain faster and more accurate results.

While anisotropic meshes offer many benefits, their generation is also much harder than the traditional isotropic meshes. Due to its wide range of applications, several classes of methods have been proposed, yet no solution is satisfying for all classes of domains and anisotropy. Moreover, while isotropic meshes and their generation are now well studied from a theoretical point of a view, almost all the algorithms on anisotropic mesh generation are heuristic. With this observation in mind, we seek to develop methods that are both provable, robust, and practical.

Contributions

The different approaches that we consider in this thesis are all based upon extending the notions of Voronoi diagrams and Delaunay triangulations to the anisotropic setting as we hope to benefit from the known results and theoretical soundness of the isotropic Delaunay triangulation and Voronoi diagram to generate anisotropic meshes with provable and practical meshing techniques. As all our methods are based upon the same structures, we dedicate a chapter to introducing the notions that will be used throughout this thesis. Our main chapters follow a logical progression, with each method taking more metric information into account to determine the connectivity and placement of points than the previous ones.

We begin with a thorough practical investigation of the framework of locally uniform anisotropic meshes, a theoretically sound meshing technique proposed by Boissonnat et al. that is based on the idea of constructing at each point a triangulation that is well adapted to the local metric. The theoretical aspect of their approach has been extensively described, but its practicality is comparatively lesser known. We detail our implementation, which is both more robust and faster than the one previously presented in the short experiment investigation of the algorithm for surfaces, investigate the role of the numerous parameters, and give some results. Limitations of the approach are then exposed, along with our attempts to address those.

In the Euclidean setting, the Delaunay triangulation of a point set can be constructed by first generating the Voronoi diagram of the point set and then computing the dual of this diagram. Anisotropic Voronoi diagrams have been considered to build anisotropic triangulations, however the dual of an anisotropic Voronoi diagram is not necessarily a valid triangulation and elements can be inverted. Different distances are possible to create such anisotropic Voronoi diagrams. We consider in Chapter II.5 the anisotropic distance proposed by Labelle and Shewchuk. Along with the introduction of their anisotropic distance, Labelle and Shewchuk presented a refinement algorithm that generates a point set for which their anisotropic Voronoi diagram has a valid dual triangulation. However, their method is limited to the setting of planar domains. We give requirements on point sets such that the dual of the anisotropic Voronoi diagram is a nice triangulation and propose a refinement algorithm to generate such point set. Our proof links anisotropic Voronoi diagrams built using the distance of Labelle and Shewchuk with the framework of locally uniform anisotropic meshes, relating along the way the concepts of quasi-cosphericity (used in locally uniform anisotropic meshes) and of protection of a point set. The second part of this chapter introduces a refinement algorithm based on the combination of the alternative point of view of the anisotropic Voronoi diagram as the restriction of a high-dimensional power diagram to a fixed paraboloidal manifold and the tangential Delaunay complex, a structure used in manifold reconstruction that is well-adapted to the high-dimensional setting. We detail our implementation, study its theoretical grounds and investigate the practicality of the algorithm.

Anisotropic Voronoi diagrams studied by previous authors compute and compare distances using a fixed metric, justifying this approximation by invoking the computational and time complexity of computing geodesics in a domain endowed with a metric field. In Chapter III.6, we introduce discrete Riemannian Voronoi diagrams, an accurate approximation of Riemannian Voronoi diagrams

that aims at palliating some of the disadvantages of using geodesics. The speed of classical geodesic computations is greatly improved by the use of a non-embedded anisotropic graph generated with the algorithm presented in Chapter I.4. The algorithm produced good results for planar and surface domains. Chapters III.7 and III.8 are devoted to the proof of the theoretical soundness of our approach first in two dimensions, and finally in any dimension.

1. State of the Art

Anisotropic mesh generation

Due to its wide array of practical applications, anisotropic mesh generation has received considerable attention and several classes of methods have been proposed. We categorize these past works into three main categories: algorithms based on the concepts of the Delaunay triangulation and the Voronoi diagram, algorithms based on an embedding of the input domain to simplify the problem, and algorithm based on the optimization of particles. The boundaries between each category is not hermetic and methods often draw from more than one of these categories. Approaches are sorted according to what we felt was the primary category.

From a historical point of view, the generation of isotropic unstructured triangulations was originally tackled using packing, and advancing front (triangles or tets are placed from the boundary and advance towards the interior of the domain) methods. Parameterization techniques are based on creating meshes in simpler domains and mapping them back to the original domain. Finally, Delaunay refinement was introduced and combined iterative insertions with the Delaunay triangulation to offer robust algorithms. Comprehensive descriptions of structured and unstructured meshing algorithms have been published by Owen [113], and Frey and George [72].

The same chronology has been roughly followed in the context of anisotropic mesh generation. Bossen and Heckbert proposed a method called pliant mesh generation that is based on the use of repulsion, smoothing, insertion, and deletion within a loop. Similarly, Vallet combined insertion, deletion, smoothing and re-meshing techniques [136].

Jiao [82] used the quadratic metric tensor of Heckbert and Garland [75] and proposed a set of edge operations to allow anisotropic adaptation of a triangular mesh of a static or evolving surface.

Particle-based methods

Modifying the position of a point is a popular technique in mesh generation and optimization [132].

Shimada proposed a series of algorithms based on the packing of bubbles with the idea that they provide good approximations of the Voronoi diagram. The bubble method can be summarized as a sequence of two steps: pack spheres (or bubbles) closely in the domain, and connect their centers. First applied to isotropic mesh generation [129], Shimada extended this idea to the anisotropic setting by considering ellipses instead of spheres [130], and to three-dimensional domains using ellipsoids [142].

Adaptive Smoothed Particle Hydrodynamics by Shapiro, anisotropic mesh adaptation.

Optimal Delaunay triangulations were introduced by Chen et al. [44, 42, 43] as the triangulation that minimizes the interpolation error $\|u - \hat{u}\|$ of a function u and its piecewise interpolation $\hat{u}$ over the mesh. The advantage of this minimization is double as it is both topological and geometrical. First applied to isotropic mesh generation using $u(x) = x^2$ [6, 134], optimal Delaunay triangulations were naturally extended to anisotropic metric fields derived from the Hessian of a convex function through the use of weighted Delaunay triangulations [97, 106]. However, as shown by Boissonnat et al. [16], a Riemannian metric field generally cannot be expressed as the Hessian of a convex function. Chen et al. [42] attempted to palliate this restriction by locally approximating the metric field with local convex functions, but their algorithm may fail to converge or achieve the desired anisotropy. Fu

et al. [73] tweaked this approach by including feedback from the neighboring vertices in the expression of the local energy, with more success.

The smoothing process that is inherent to any method based on optimization is a double-edged sword: while it provides visually pleasant meshes and makes methods robust to geometric or metric noise, it can also result in the loss of anisotropy in regions of a domain where the metric field is changing rapidly as the result of multiple smoothing creates an isotropic mesh. Furthermore, smoothing is often badly controlled in the case of piecewise inputs, with the metric field on a polygonal face dribbling on the adjacent faces. Finally, none of these methods can prove that their optimization process converges or that the mesh is an anisotropic mesh of good quality.

Delaunay refinement

The good properties of the Delaunay triangulations have made it a powerful tool in mesh generation.

In early isotropic meshing techniques, Delaunay triangulations were constructed from existing point sets, generated through other means. Incorporating the Delaunay structure of the point set in the generation of the point set was pioneered in 2D by Chew [52] and Ruppert [123]. They proposed to iteratively insert the circumcenters of triangles that do not fit a given set of criteria, such as the size or shape of elements. This technique allowed to generate meshes with lower bounds on the smallest angle of triangles and was then extended to 3D domains. In 3D, although no guarantee can be made on the dihedral angles of simplices, the radius-edge ratio of tetrahedra can be shown to be bounded.

An efficient algorithm to insert a vertex in an isotropic Delaunay triangulation, now known as the Bowyer-Watson algorithm, was proposed simultaneously and independently by Bowyer [34] and Watson [138]. The simplices whose Delaunay ball contains the simplex are collected and removed from the triangulation, which forms a cavity. Linking the refinement point to the vertices of the border of the cavity creates the new Delaunay simplices of the triangulation.

In the context of anisotropy, the generation of points was also originally done independently from the construction of the triangulation, using for example anisotropic quad trees. Mavriplis first considered the idea of stretched Delaunay methods [102, 135, 103] and using nodes generated from an anisotropic advancing front technique; the connectivity is set by first constructing a large isotropic mesh and then inserting vertices with the Bowyer-Watson algorithm adapted to the anisotropic setting. The stretching of the space is obtained by computing gradients of the solution. Good results were achieved, but the swapping techniques employed do not extend nicely to higher-dimensional settings.

Borouchaki et al. [33] formalized the approach of stretching spaces of Mavriplis through the use of Riemannian metric tensors and introduced the anisotropic Delaunay kernel, their anisotropic version of an anisotropic Bowyer Watson algorithm. Along with this new insertion algorithm, they introduced a Delaunay refinement algorithm based on edge swapping, merging and splitting techniques to generate meshes whose edges' lengths are close to 1 in the metric at each of their endpoints. Many developments have sprouted from this approach: 3D mesh generation [60], periodic anisotropic mesh generation [61], metric-orthogonal mesh generation [101].

While these algorithms produce good results and have seen much use in the context of computational fluid dynamics [4], theoretical results are limited for these algorithms and there are no guarantees on either the termination or the robustness of algorithms, nor on the quality of the elements produced by these techniques.

A theoretically sound approach to anisotropic Delaunay triangulations was proposed by Boissonnat et al. [30], who introduced the framework of locally uniform anisotropic meshes. In their algorithm, the star of each vertex v is composed of simplices that are Delaunay for the metric at v. Each star is built independently and the stars are stitched together in the hope of creating an anisotropic mesh.

The star structure was first introduced by Shewchuk [128] to handle moving vertices in finite element meshes and considering stretched stars was first proposed by Schoen [126]. Two stretched stars may be combinatorially incompatible, a configuration called an *inconsistency*. Boissonnat et al. proved that inconsistencies can be resolved by inserting Steiner points, yielding an anisotropic triangulation. The algorithm works in any dimension, can handle complex geometries and provides guarantees on the quality of the simplices of the triangulation. This algorithm will be investigated thoroughly in Chapter I.4.

Voronoi diagrams

The well-known duality between the Euclidean Voronoi diagram and its associated Delaunay triangulation has inspired authors to compute anisotropic Voronoi diagrams, with the hope of obtaining a dual anisotropic triangulation.

The approaches of Labelle and Shewchuk [89] and Du and Wang [64] aim at approximating the geodesic distance between a seed and a point of the domain by considering that the metric is constant and equal to the metric at the seed (in the case of Labelle and Shewchuk [89]) or at the point (in the case of Du and Wang [64]). Contrary to the isotropic setting, the dual of an anisotropic Voronoi diagram is not necessarily a triangulation and inverted elements can be present in the dual triangulation. The algorithms were initially introduced for two-dimensional (Labelle and Shewchuk [89]) and surface (Du and Wang [64]) domains and have since then been studied and extended by various authors. Canas and Gortler [38] proved that point sets could be generated (in 2D) such that the dual of the Voronoi diagram obtained with the distance of Du and Wang is a triangulation. The approach of Labelle and Shewchuk was shown to be theoretically sound in 2D, but the approach of the proof does not extend to higher dimensions. This result was extended to surfaces by Cheng et al. [50] by locally approximating the surface with a plane and then using a density argument similar to the proof of Canas and Gortler [38].

Centroidal Voronoi tessellations, which are Voronoi diagrams for which the seeds are the centers of mass of their associated Voronoi cell, are known to create elements of good quality. The famous Lloyd algorithm [100] iteratively moves the seeds to the center of mass of their respective cell and recomputes the Voronoi diagram of this new seed set. This algorithm was modified to be used in the anisotropic Voronoi diagram of Du and Wang, but the process is computationally expensive.

Embedding and parametrization

The generation of anisotropic meshes being a difficult problem, various authors have tried to simplify the problem by preemptively transforming the input domain to a target domain that is easier to study and mesh. The connectivity of a mesh generated in this simpler setting is then transported back to the input point set, creating – if the deformation is well chosen – a triangulation of the input with elements stretched as desired. Although isometric embeddings are theoretically usable and would have the benefit of translating the problem to the familiar setting of isotropic mesh generation, but they are not necessarily the optimal solution. Indeed, the dimension of the embedding space is often large and due to the curse of dimensionality, traditional methods of mesh generation cannot efficiently be applied. Alternative methods must thus be considered to compute both the embedding and the generation of a mesh in the deformed space.

Lévy and Bonneel [92] proposed to lift an input surface mesh embedded in a 3D space by adding to the coordinates of the vertices the coordinates of the normal to the surface, hence obtaining six-dimensional points. They then compute an high-dimensional Euclidean centroidal Voronoi diagram, made efficient through the use of Voronoi Parallel Linear Enumeration. Unfortunately, no control is

given to the user over the choice of the metric field. To overcome this problem, Zhong et al. [144] incorporated metric information in their high-dimensional repulsive vertex-repositioning technique, al Nivoliers et al. [110] added features.

While these works claim to produce curvature-adapted anisotropic meshes, the results are often closer to an isotropic mesh whose features (sharp edges) have been meshed anisotropically, which does not provide any of the benefits that a curvature-based metric field offers. For example, meshes of cylinder-shaped parts of the input have isotropic elements on the bases and side, with anisotropic elements only appearing along the (circular) border of the bases. While theses meshes are certainly anisotropic, they do not stretch or even follow the principal curvatures of the surface and thus do not help providing a better approximation for a lesser amount of vertices as a proper curvature-based mesh would [56].

Zhong et al. [145] proposed to use a conformal parameterization of the domain upon which they compute a weighted centroidal Voronoi diagram whose connectivity is mapped back to the original point set. Their approach is however slow and the dimension of the embedding is unknown, requiring a tedious incremental search.

None of these methods can prove that the result is a triangulation in the lower dimension and often require post-processing such as the use of hole filling techniques.

Quad and polygonal anisotropic mesh generation

Anisotropic mesh generation is not restricted to simplicial meshes and it is interesting to consider what has been done for quadrangular or polygonal meshes as one can build a correspondence between quadrangular and triangular meshes, and between hexahedral and tetrahedral meshes, the anisotropic mesh generation techniques for non-simplicial meshes are relatively similar to those presented above: packing methods, optimization, etc.

Alliez et al. [5] proposed an algorithm based on the idea of following curvature lines as done in sketches to produce anisotropic quad-dominant meshes of surface. While giving good result, this method is heuristic, relies on a heavy parameterization of the domain and a produces a large number of polygonal elements, making it difficult to use in scientific simulations. The latter issue was remedied by Marinov and Kobbelt who extended the technique as to not rely on a global parameterization of the mesh, but the anisotropy poorly conforms to the curvature.

Embedding techniques have also been considered in the context of quadrangular mesh generation, for example by Panozzo et al. [114], who warped an input frame field on a surface into a cross field without increasing the dimension. Consequently, the surface contains self intersections, but they can nevertheless create an isotropic quad mesh of the deformed surface.

2. Background

The different approaches to anisotropic mesh generation considered in this thesis share a common core of notions: anisotropy is expressed through Riemannian metric tensors, final meshes are simplicial complexes, and all our algorithms are built upon the concepts of Voronoi diagrams and Delaunay triangulations. In this chapter, we recall the definitions of objects and structures that will be regularly used in the following chapters.

Metrics and metric fields are introduced to quantify the desired anisotropy over a domain. We detail basic operations such as the interpolation of metrics and the smoothing of metric fields. The concept of distortion, which was introduced by Labelle and Shewchuk [89] and allows to measure metric variation, is recalled.

Riemannian geometry is found in all chapters, and we thus present essential notions such as geodesic path, geodesic distance, and exponential map.

The Voronoi diagram and the Delaunay triangulation are the basis of all our work. In Chapter I.4, we will study Delaunay triangulations computed in an anisotropic metric. In Chapter II.5, we prove results on anisotropic Voronoi diagrams and adapt the tangential complex, a structure based on the Delaunay triangulation, to generate anisotropic meshes. Finally, in Chapter III.6, we introduce a practical approach to the computation of Riemannian Voronoi diagrams. We thus recall the definitions of Voronoi diagram and Delaunay triangulation in the general setting of Riemannian geometry, but also in the particular case of the Euclidean distance.

Lastly, the definition of nets, which are point sets with assumptions on density and sparsity, is given. Nets are constantly used throughout this thesis to construct structures with guarantees on the quality of the elements. Alongside nets, the concept of protection of point set is introduced. Power protection is a specific assumption made on Delaunay triangulations, which allows us to deduce another set of quality guarantees on the elements of the triangulation.

Contents

2.1 Basic notions

We usually consider an n-dimensional (open) domain Ω in $\mathbb{R}^n$, or an m-manifold $\mathcal{M}$ embedded in the n-dimensional Euclidean space $\mathbb{R}^n$. Euclidean distances are determined by the standard norm, $\|\cdot\|_{\mathbb{E}}$. We refer to the distance between two points p and q as $\|q-p\|$ or $d(p,q)$. A *ball* $B(c,r)$ is the set of points x of $\mathbb{R}^n$ such that $d(c,x) < r$. The boundary of a set X is denoted by ∂X and its affine hull by $\mathrm{Aff}(X)$. The cardinality of a finite set $\mathcal{P}$ is $\#(\mathcal{P})$ or $|\mathcal{P}|$. A *bijection* is a one-to-one correspondence between two sets. A *homeomorphism* is a continuous bijection whose inverse is also continuous. A *diffeomorphism* is a differentiable homeomorphism whose inverse is also differentiable.

2.2 Metric tensor

The adaptation of meshes based on a metric tensor was introduced by Borouchaki et al. [33, 59, 76] and has imposed itself as the standard way to describe and prescribe anisotropy. A *metric tensor*, or simply a *metric*, in $\mathbb{R}^n$ is defined by a symmetric positive definite (SPD) quadratic form, represented by a $n \times n$ matrix G. The quadratic form defines an inner product

$$\langle u, v\rangle_G = u^t G v = \langle u, Gv\rangle,$$

and the norm of a vector u with respect to G is then given by

$$\|u\|_G = \sqrt{\langle u, u\rangle} = \sqrt{u^t G u}.$$

Using this norm, the distance between two points p and q in Ω can be measured *in the metric* G as

$$d_G(x,y) = \|x-y\|_G = \sqrt{(x-y)^t G (x-y)}.$$

This definition can be naturally extended to higher-dimensional G-volume measures by integration.

Given a metric G, the G-sphere $S_G(c,r)$ with center c and radius r is the set of points p such that $d_G(c,p) = r$, and, likewise, the closed G-ball $B_G(c,r)$ is defined as the set of points p such that $d_G(c,p) \le r$. A metric G can be visualized by its unit sphere $S_G(c,1)$, which is an ellipsoid in $\mathbb{R}^n$. Indeed, since G is a SPD matrix, it is diagonalizable as $G = O^T D O$, where O is an orthogonal matrix and D is a diagonal matrix. The equation of the unit sphere is

$$x^T G x = 1 \iff x^T O^T D O x = (Ox)^T D (Ox) = \sum_{i=1}^{n} ((Ox)_i)^2 \lambda_i = 1$$

One recognizes the equation of an ellipsoid with semi-axis lengths $1/\sqrt{\lambda_i}$. Since O is an orthonormal matrix, it can also be seen as a rotation matrix and the unit sphere in the metric is thus an ellipsoid in $\mathbb{R}^n$ whose axes' directions are given by the eigenvectors $\{v_i\}$ of G and whose semi-axis lengths are equal to $1/\sqrt{\lambda_i}$ in the direction v_i, where λ_i is the eigenvalue corresponding to v_i. Incidentally, this construction proves that there is a natural bijection between ellipsoids and metrics.

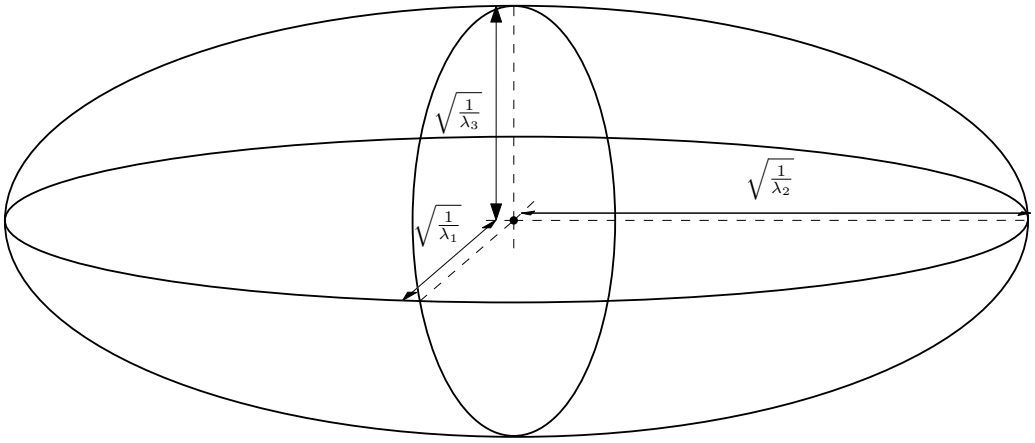

Figure 2.1: Representation of a 3D metric with eigenvalues λ_1, λ_2 and λ_3 as an ellipsoid.

Metric transformation Given a symmetric positive definite matrix G, we denote by F any matrix such that $\det(F) > 0$ and $F^tF = G$. The matrix F is called a *square root* of G. The square root of a matrix is not uniquely defined: the Cholesky decomposition provides, for example, an upper triangular F, but it is also possible to consider the diagonalization of G as O^TDO, where O is an orthonormal matrix and D is a diagonal matrix; the square root is then taken as $F = O^T\sqrt{D}O$. The latter definition is more natural than other decompositions since $\sqrt{D}$ is canonically defined and F is then also a symmetric definite positive matrix, with the same eigenvectors as G. Regardless of the method chosen to compute the square root of a metric, we assume that it is fixed and identical for all the metrics considered.

The square root F offers a linear bijective transformation between the Euclidean space and a metric space, noting that

$$d_G(x,y) = \sqrt{(x-y)^tF^tF(x-y)} = \|F(x-y)\| = \|Fx - Fy\|\,,$$

where $\|\cdot\|$ stands for the Euclidean norm. Thus, the metric distance between x and y in Euclidean space can be seen as the Euclidean distance between two *transformed* points Fx and Fy living in the metric space of G.

2.2.1 Metric field

The desired anisotropy ratio and directions are generally not constant over the domain and a single metric is thus not sufficient to completely express anisotropy. A *metric field* g, defined over a domain $\Omega \subset \mathbb{R}^n$, associates a metric $g(p) = G_p = F_p^tF_p$ to any point p in Ω. When the map $g : p \mapsto G_p$ is constant, the metric field is said to be *uniform*.

In the following, we shall replace the subscript G_p by p to avoid double subscripts, and simply write X_p instead of X_{G_p}. Figure 2.2 shows two metric fields defined over a planar domain, the Euclidean metric (left), and a radial metric field (right) whose definition is detailed in Section A.6 in Appendix A.

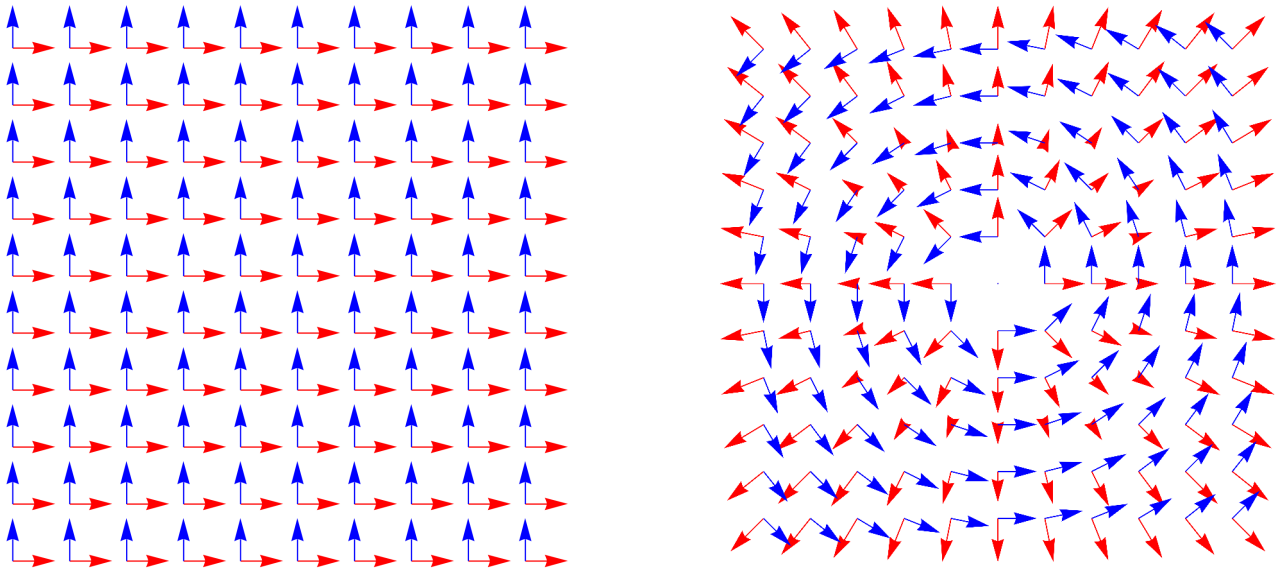

Figure 2.2: Isotropic (left) and anisotropic (right) 2D metric fields. The metric at each sample point is expressed by its (2) eigenvectors, each scaled by its respective eigenvalue.

In the following, we assume that $\Omega \subset \mathbb{R}^n$ is a compact domain endowed with a continuous Lipschitz metric field g.

Ambient and intrinsic metric fields

If the domain is a m-manifold $\mathcal{M}$ embedded in an ambient space of dimension n, metric fields can be defined in the ambient space (the metric is given by a $n \times n$ SPD matrix), but also intrinsically on the manifold (the metric at $p \in \mathcal{M}$ is then represented by a $m \times m$ SPD matrix whose eigenvectors live in the tangent space $T_p\mathcal{M}$).

Since the intersection of an ellipsoid with an affine space is a lower-dimensional ellipsoid [87], it is possible to go from the ambient to the intrinsic setting (but not conversely).

2.2.2 Distortion

The notions of metrics and metric fields are introduced to convey the stretching of spaces. To create a solid theoretical framework, it is required to be able to express how differently two metrics see distances and geometrical objects. For this purpose, Labelle and Shewchuk [89] introduced the concept of *distortion* between two metrics. We recall here their definition. Properties and limitations as well as a new alternative that remedy those shortcomings are proposed in Section 3.1 in Chapter 3.

Defining distortion: first approach

Labelle and Shewchuk [89] introduced the concept of *distortion* between two points p and q of Ω as

$$\psi(p,q) = \psi(G_p, G_q) = \max\left\{\left\|F_p F_q^{-1}\right\|, \left\|F_q F_p^{-1}\right\|\right\},$$

where $\|\cdot\|$ is the Euclidean matrix norm, that is $\|M\| = \sup_{x\in\mathbb{R}^n} \frac{\|Mx\|}{\|x\|}$. Observe that $\psi(G_p, G_q) \geq 1$ and $\psi(G_p, G_q) = 1$ when $G_p = G_q$.

A fundamental property of the distortion is to relate the two distances d_{G_p} and d_{G_q}. Specifically, for any pair x, y of points, we have

$$\frac{1}{\psi(p,q)}\, d_{G_p}(x,y) \leq d_{G_q}(x,y) \leq \psi(p,q)\, d_{G_p}(x,y). \tag{2.1}$$

Indeed,

$$d_p(x,y) = \left\|F_p(x-y)\right\| = \left\|F_p F_q^{-1} F_q(x-y)\right\| \leq \left\|F_p F_q^{-1}\right\| \left\|F_q(x-y)\right\| \leq \psi(G_p, G_q)\, d_q(x,y),$$

and the other inequality is obtained similarly.

Remark 2.2.1 *The distortion provides a measure of difference between metrics, but there is no direct link between distortion and anisotropy. For example, it is possible for two metrics to have a low distortion but to encode very large anisotropy. The extreme case is that of a uniform metric field where the distortion is* 1 *between any two points but the anisotropy ratio can be as large as desired.*

Despite being easy to define and compute, this definition lacks a geometrical understanding. Additionally, the guarantee of good behavior of algorithms will often come through bounds on the distortion between neighboring sample points. While it is possible to extract a sampling condition from a distortion bound, it requires unreasonable assumptions on the metric field, see Section 3.1 in Chapter 3. For these reasons, we will introduce a new alternate definition of distortion in Section 3.1 in Chapter 3.

Distortion of multiple points The distortion between two points can naturally be extended to sets of points, domains or objects as the maximum distortion between any two points of the structure considered. For example, the distortion of a simplicial complex $\mathcal{K}$ is given by

$$\psi(\mathcal{K}) = \max_{p_i, p_j \in \mathcal{K}} \psi\left(p_i, p_j\right).$$

2.2.3 Operations on metric fields

We detail in this section some basic operations on metric fields that are sometimes required in practice.

Interpolation of metrics

Although our definition of metric field is continuous, a metric field is not always provided continuously in input. For example, the computation of a metric field derived from the Hessian of the solution of a partial differential equation or from the numerical estimation of the curvature of a piecewise domain usually yields a metric field that is known only at the vertices of the input mesh. To provide knowledge of the metric field at new or displaced vertices, we must be able to extend a discrete metric field continuously over the complete domain.

The estimation of the metric field at a query point p on Ω is done by locating p in the domain and interpolating the metrics of neighboring vertices. These neighboring vertices are either known ahead of the interpolation or computed through (isotropic or anisotropic) nearest neighbor search algorithms. The weight of a point is usually based on the anisotropic distance between this point and the query point. If the input domain is a simplicial complex (see Section 2.5), fast algorithms are known [58] to locate queries. If the neighbors are defined as the vertices of the simplex that contains the query, barycentric coordinates are a natural choice for weights.

Once the neighbors and their respective weights are known, various interpolation techniques can be used. We now recall some of these techniques, along with their properties. Further details can be found in the works of Kindlmann [86].

Linear The linear interpolation of a family of matrices $\{G_i\}_{i=1...N}$ with associated (positive) weights $\{\omega_i\}_{i=1...N}$ such that $\sum_i \omega_i = 1$ is defined by

$$G = \sum_{i=1}^{N} \omega_i G_i$$

G is a positive combination of symmetric tensors and therefore has no negative eigenvalues. This property can be proved by contradiction. Assume that G has some negative eigenvalues. Then there exists a vector x that satisfies $x^t G x = \sum_{i=1...N} \omega_i x^t G_i x < 0$. Since all G_i are symmetric tensors, $x^t G_i x \geq 0$ and, since according to the definition, $\omega_i \geq 0$, we have $x^t G x \geq 0$, which provides a contradiction.

The simplicity of the definition and the implementation of the linear interpolation comes at the cost of a "swelling effect" on the interpolated metric: the determinant of the interpolated tensor is greater than those of the interpolated metric tensors.

Ellipse intersections Borouchaki et al. [33] proposed various types of interpolations, and in particular used the geometric representation of metrics as ellipsoids to define the intersection of ellipses. The intersection of two metrics is given by the largest ellipsoid that can be fitted in the geometrical intersection of the two ellipsoids. From an analytical point of this view, this can be computed through simultaneous reduction of the metric tensors. It is however non-commutative and is difficult to use with multiple metrics.

Affine-Euclidean Batchelor et al. [13] sought to create a rigorous framework for the interpolation of metric tensors. They define the interpolation between two metric tensors by considering the geodesic between these two metrics in the space of metric tensors. When interpolating between more than 2 metrics, this requires solving numerically a system, making it inexact and expensive to compute.

Log-Euclidean The Log-Euclidean Fréchet mean was introduced by Arsigny et al. [10]. Contrary to the geodesic-loxodrome and affine-Euclidean interpolations, it has the advantage of being easy and fast to compute.

For a family of metrics $\{G_i\}_{i=1...N}$ and associated weights $\{\omega_i\}_{i=1...N}$ such that $\sum_i \omega_i = 1$, the interpolated metric is given by:

$$G = \exp\left(\sum_{i=1}^{n} \omega_i \ln(G_i)\right),$$

where the matrix exponential is given by the series

$$\exp(M) = \sum_{k=0}^{\infty} \frac{M^k}{k!}.$$

If M is diagonalizable with decomposition $M = P^{-1}\Delta P$, where $\Delta = \mathrm{Diag}\{\lambda_1, \ldots, \lambda_n\}$, then

$$\left(P_L^{-1}\Delta_L P_L\right)^k = \underbrace{\left(P_L^{-1}\Delta_L P_L\right)\left(P_L^{-1}\Delta_L P_L\right)\ldots\left(P_L^{-1}\Delta_L P_L\right)\left(P_L^{-1}\Delta_L P_L\right)}_{k} = P_L^{-1}\Delta_L^k P_L,$$

and $M^k = P^{-1}\Delta^k P$. The exponential is therefore directly applied to the entries of the diagonal matrix:

$$\exp(M) = P^{-1}\exp(\Delta)P = P^{-1}\begin{pmatrix} \mathrm{e}^{\lambda_1} & & \\ & \ddots & \\ & & \mathrm{e}^{\lambda_n} \end{pmatrix} P. \tag{2.2}$$

The logarithm of a matrix is defined as the inverse operator of the exponential, and, similarly, we have for diagonalizable matrices that

$$\ln(M) = P^{-1} \ln(\Delta) P.$$

The matrix $L = \sum_{i=1}^{n} \omega_i \ln(G_i)$ is a well-defined metric tensor. It is symmetric, so it is diagonalizable with real eigenvalues as $L = P_L^{-1} \Delta_L P_L$. As for Equation 2.2, we have

$$G = \exp(L) = P_L^{-1} \exp(\Delta_L) P_L.$$

The interpolated G is symmetric positive definite, and is thus a metric.

Note that the Log-Euclidean interpolation reduces to a geometric mean of the family of matrices if the matrices commute:

$$\exp\left(\sum_{i=1}^{n} \omega_i \ln(M_i)\right) = \prod_{i=1}^{n} M_i^{\omega_i}$$

Figure 2.3 the Log-Euclidean interpolation of three random metric tensors defined at the vertices of a triangle and their interpolation at various points of the triangle.

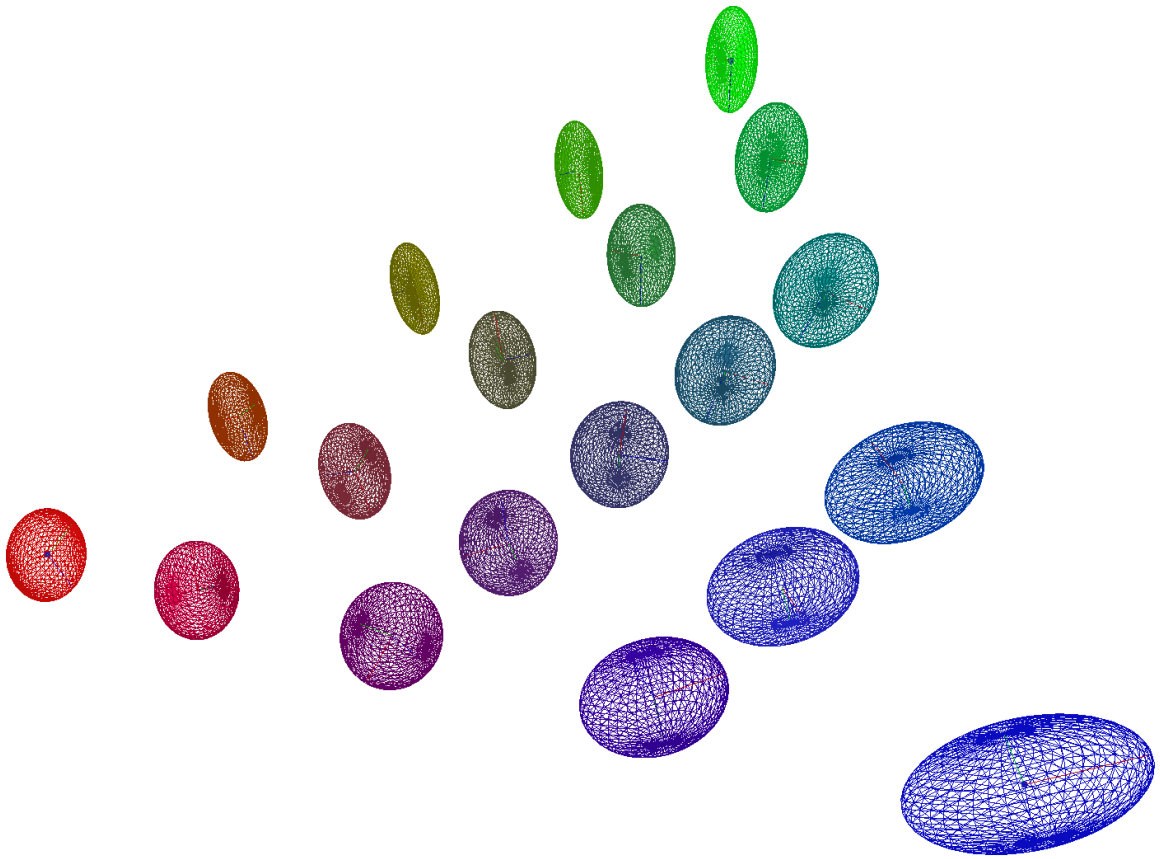

Figure 2.3: Log-Euclidean interpolation of three random metric tensors (colored red, green and blue).

Conveniently, the Log-Euclidean interpolation can be directly computed on the square root F_p. Indeed, the interpolated of the family $\{F_i\}$ of square roots of $\{G_i\}$ with the weights ω_i is given by

$$F = \exp\left(\sum_i \omega_i \ln\left(F_i\right)\right).$$

We have on the other hand that

$$\begin{aligned} G &= \exp\left(\sum_i \omega_i \ln\left(F_i^t F_i\right)\right) \\ &= \exp\left(\sum_i \omega_i \ln\left(F_i^t\right) + \sum_i \omega_i \ln\left(F_i\right)\right) \text{ since } F_i \text{ and } F_i^t \text{ commute} \\ &= \exp\left(\sum_i \omega_i \ln\left(F_i^t\right)\right) \exp\left(\sum_i \omega_i \ln\left(F_i\right)\right) \\ &= \left(\exp\left(\sum_i \omega_i \ln\left(F_i\right)\right)\right)^t \exp\left(\sum_i \omega_i \ln\left(F_i\right)\right) = F^t F, \end{aligned}$$

using $\ln(M^t) = \ln(M)^t$ and $\exp(M^t) = \exp(M)^t$. This is particularly useful in the context of locally uniform anisotropic meshes (see Chapter I.4), where we make extensive use of the stretching transformation provided by F_p.

Metric interpolation and distortion The distortion between metrics plays a central role in the generation of anisotropic meshes. When judging the quality of an anisotropic mesh, one evaluates the shape of its elements, by for example computing its smallest angle. Naturally, this evaluation must be done in the metric and we usually evaluate the shape of an element in the metrics of all of its vertices. Therefore, the lower the distortion, the easier it is to create anisotropic elements that conform to the metric field.

Not only is the Log-Euclidean the fastest and easiest method to compute among those described above, but it is also the interpolation technique that produces – on average – the lowest distortion when interpolating the metrics of the vertices of an element and the refinement point of that element (which is generally its circumcenter as we study Delaunay complexes, see Section 2.6).

Geodesic-loxodrome Kindlmann et al. [86] introduced another form of interpolation as a shortest path in the tensor space: they impose that the path between two metrics should keep the same bearing, that is the same shape. The interpolation between both metrics is thus a geodesic, but only on the tensor space of metrics with the same shape. This results in a finer interpolation between tensors and especially avoids the appearance of isotropic tensors during the interpolation of two very different anisotropic tensors, which might sometimes happen in other methods. However, the interpolation is obtained through numerical computations, and is more complex than the Log-Euclidean and thus was discarded.

Smoothing of metric fields

In practice, noise is often present in a metric field and can have different sources: a geometrically imprecise input, inaccuracies in the solution evaluation, or simply a poor numerical evaluation of the metric field (for example, in the computation of the curvature). Smoothing a metric field correspond to (closely) approximating it by a more regular metric field that leaves out highly local and rapid variations. Our smoothing process of a metric field is described below.

Denote N_p a chosen set of neighboring vertices of p (for example the geodesic k-nearest vertices of p in the domain). The smoothed metric tensor $\widetilde{G}_p$ at a vertex p is computed from G_p and from the

metric tensors G_q at neighboring vertices $q \in N_p$, using the interpolation operator previously defined:

$$\widetilde{G_p} = \exp\left(\omega(p,p)\ln(G_p) + \sum_{q \in N_p} \omega(p,q)\ln(G_w)\right)$$

with

$$\omega(p,q) = \mu(p,q) \Big/ \left(\mu(p,p) + \sum_{q \in N_p} \mu(p,q)\right)$$

where $\mu(p,q) = e^{-\|p-q\|^2/r^2}$ and r is a parameter that controls the locality of the smoothing. By default, r can be computed by averaging the distance to the vertices in N_p. The interpolant $\widetilde{G_p}$ is a metric tensor by construction from the Log-Euclidean interpolation.

Remark 2.2.2 *Our smoothing operation is based on the Log-Euclidean interpolation (defined in Section 2.2.3), but any other interpolation could be also used to define another smoothing operator.*

2.3 Riemannian geometry

Ideally, the computation of the anisotropic distance between two points would use the metric field information continuously. From a theoretical point of view, this corresponds to the framework of Riemannian geometry. This theoretical setting will be regularly used and we thus recall some definitions of differential geometry.

A *path* between two points x, y of Ω is a function $\gamma : [0,1] \to \Omega$ such that $\gamma(0) = x$ and $\gamma(1) = y$. The *Riemannian length* of a path γ is defined as

$$\ell_g(\gamma) = \int_0^1 \sqrt{\dot\gamma(t)^t g(\gamma(t))\dot\gamma(t)}\, dt.$$

A *geodesic* between x and y is a locally length-minimizing curve. Let $\Gamma(x,y)$ be the set of paths from x to y on Ω: $\Gamma(x,y) = \{\gamma : [0,1] \to \Omega \mid \gamma(0) = x \text{ and } \gamma(1) = y\}$, we define the *Riemannian geodesic distance* between two points as given by

$$d_g(x,y) = \min_{\gamma \in \Gamma(x,y)} \ell_g(\gamma).$$

The *shortest path* between x and y is given by $\arg\min d_g(x,y)$. In the following, we replace the subscript d_g by simply g to avoid double subscripts, and write X_g instead of X_{d_g}.

Remark 2.3.1 *We define the geodesic distance $d_g(x,y)$ between two points as the length of the shortest path between x and y and not the length of any geodesic between x and y.*

Most geometrical objects previously described can be generalized using the geodesic distance. For example, the geodesic (closed) ball centered on $p \in \Omega$ and of radius r is given by $B_g(p,r) = \{x \in \Omega \mid d_g(p,x) \leq r\}$. Equation 2.1, which relates distortion and distances for two metrics, can be extended to relate geodesic distances with respect to two metric fields g and g' (see Lemma 3.1.4).

Let v be a vector living in the tangent space $T_p\mathcal{M}$. From the unique geodesic γ satisfying $\gamma(0) = p$ with initial tangent vector $\dot\gamma$, one can define the *exponential map* $\exp(v) : T_p\mathcal{M} \to \mathcal{M}$ by $\exp(v) = \gamma(1)$. If geodesics can be extended infinitely, the manifold is said to be *geodesically complete*. When $\mathcal{M}$ is geodesically complete, the exponential map is defined over the whole tangent space, but this does not

imply that the exponential map is a homeomorphism over the whole tangent space. The *injectivity radius* at a point p of $\mathcal{M}$ is the largest radius for which the exponential map at p is a diffeomorphism. The injectivity radius of the manifold $\mathcal{M}$ is defined by the infimum of the injectivity radii at all points. For any $p \in \mathcal{M}$ and for a two-dimensional subspace $H \subseteq T_p\mathcal{M}$, we define the *sectional curvature* at p for H as the Gaussian curvature at p of the surface $\exp_p(H)$.

In the theoretical studies of our algorithm, we will generally assume that the domain $\mathcal{M}$ is geodesically complete and has no boundaries in order to avoid punctures.

2.4 Voronoi diagram

The *Voronoi diagram* of a set of points $\mathcal{P} = \{p_i\}$ with respect to a distance function d, denoted by $\mathrm{Vor}_d(\mathcal{P})$, is a partition of a domain Ω in a set of regions $\{V_d(p_i)\}$. The region $\mathrm{V}_d(p_i)$ is called the *Voronoi cell* of the *seed* p_i and is defined by

$$\mathrm{V}_d(p_i) = \{x \in \Omega \mid d(p_i, x) \leq d(p_j, x), \forall p_j \in \mathcal{P} \setminus p_i\}.$$

For a fixed set of seeds, different distances naturally produce different Voronoi diagrams.

We will make regular use of a number of geometrical objects related to Voronoi diagrams. A *Voronoi face* is the intersection of a set of Voronoi cells. Specifically, a Voronoi k-face is a face of dimension k and is the intersection of $n+1-k$ Voronoi cells. The *boundary* of a Voronoi k-face W is the union of the $(k-1)$-faces that are incident to W. If the dimension of the face is 0 or 1, the terms *Voronoi vertex* and *Voronoi edge* are respectively employed. The *bisector* of two seeds p and q is the set $\mathrm{BS}(p, q) = \{x \in \Omega, d(p, x) = d(q, x)\}$. An *orphan* is a region of a cell that is not connected to its seed.

Voronoi diagrams have been intensively studied for various metrics, such as the Euclidean metric, the Manhattan and Moscow metrics [112], or L^p metrics [93]. We focus on the Euclidean Voronoi diagram, defined using the Euclidean distance, and the Riemannian Voronoi diagram, defined using the geodesic distance. The former is presented below, and the latter is detailed in Section 2.8.

2.4.1 Euclidean Voronoi diagram

When the Euclidean distance $d_{\mathbb{E}}$ is used, the (Euclidean) Voronoi cell V_i of p_i is defined by

$$\mathrm{V}_{\mathbb{E}}(p_i) = \{x \in \Omega \mid \|p_i - x\| \leq \|p_j - x\|, \forall p_j \in \mathcal{P} \setminus p_i\}.$$

The subspace of points equidistant to two seeds p, q with respect to $d_{\mathbb{E}}$ is given by

$$\mathrm{BS}(p, q) = \{x \in \Omega \mid d_{\mathbb{E}}(p, x) = d_{\mathbb{E}}(q, x)\}.$$

Since

$$|x - p|^2 = |x - q|^2 \iff 2x(p - q) + q^2 - p^2 = 0,$$

the bisector of two seeds in an Euclidean Voronoi diagram is an hyperplane and a Voronoi cell is thus the intersection of a set of half-spaces forming a (possibly unbounded) convex polytope.

2.5 Simplices and complexes

Before defining Delaunay triangulations, we introduce the more general concept of simplicial complexes. Since the standard definition of a simplex as the convex hull of a set of points does not extend

well to the Riemannian setting (see Dyer [65]), we approach these definitions from a more abstract point of view.

A *simplex* σ is defined as a non-empty finite set. The elements of σ are called the *vertices* of σ and the set of vertices of σ is noted $\mathrm{Vert}(\sigma)$. The *dimension* of σ is given by $\dim \sigma = \#(\sigma) - 1$, and a j-simplex refers to a simplex of dimension j. If a simplex σ is a subset of τ, we say that it is a *face* of τ, and write $\sigma \leq \tau$. A 1-dimensional face is called an *edge* and a *facet* of τ is a face σ with $\dim \sigma = \dim \tau - 1$.

For any vertex $p \in \sigma$, the face *opposite* to p in σ is the face determined by the other vertices of σ, and is denoted by σ_p. If σ is a j-simplex, and p is not a vertex of σ, we may construct a $(j+1)$-simplex $\tau = p * \sigma$, called the *join* of p and σ as the simplex defined by p and the vertices of σ.

The *length* of an edge is the distance between its vertices. A *circumscribing ball* for a simplex σ is any n-dimensional ball that contains the vertices of σ on its boundary. If σ admits a circumscribing ball, then it has a circumcenter, $C(\sigma)$, which is the center of the unique smallest circumscribing ball for σ. The radius of this ball is the *circumradius* of σ, denoted by $R(\sigma)$. The *height* of p in σ is $D(p, \sigma) = d(p, \mathrm{Aff}(\sigma_p))$. The *dihedral angle* between two facets is the angle between their two supporting planes. If σ is a j-simplex with $j \geq 2$, then for any two vertices p, $q \in \sigma$, the dihedral angle between σ_p and σ_q defines an equality between ratios of heights (see Figure 2.4).

$$\sin \angle(\mathrm{Aff}(\sigma_p), \mathrm{Aff}(\sigma_q)) = \frac{D(p, \sigma)}{D(p, \sigma_q)} = \frac{D(q, \sigma)}{D(q, \sigma_p)}.$$

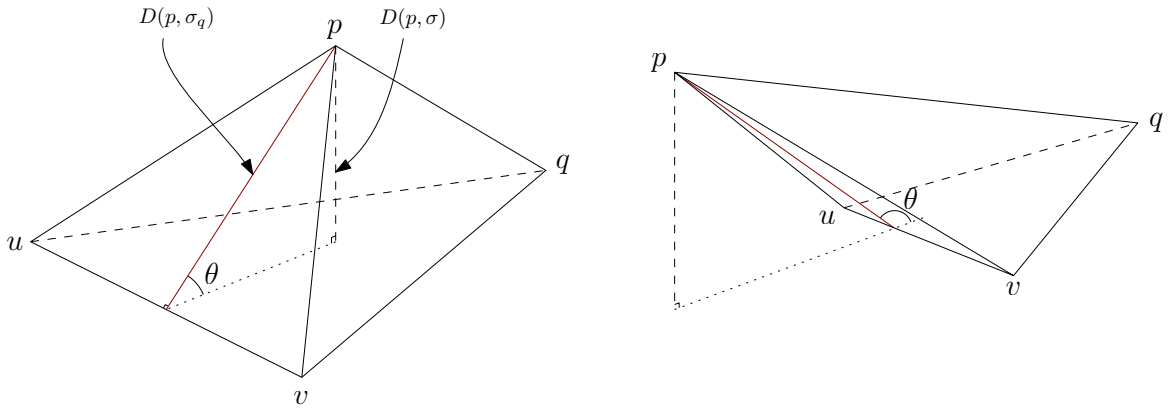

Figure 2.4: Acute and obtuse dihedral angles

2.5.1 Simplicial complexes

Simplicial complexes form the underlying framework of Delaunay triangulations. An *abstract simplicial complex* is a set $\mathcal{K}$ of simplices such that if $\sigma \in \mathcal{K}$, then all the faces of σ also belong to $\mathcal{K}$. The union of the vertices of all the simplices of $\mathcal{K}$ is called the *vertex set* of K. The dimension of a complex is the largest dimension of any of its simplices. A subset $L \subseteq \mathcal{K}$ is a *subcomplex* of $\mathcal{K}$ if it is also a complex. Two simplices are *adjacent* if they share a face and *incident* if one is a face of the other. If a simplex in $\mathcal{K}$ is not a face of a larger simplex, we say that it is *maximal*. If all the maximal simplices in a complex $\mathcal{K}$ of dimension n have dimension n, then the simplicial complex is *pure*. The *star* of a vertex p in a complex $\mathcal{K}$ is the subcomplex S formed by set of simplices that are incident to p. The *link* of a vertex p is the union of the simplices opposite of p in S_p.

A *geometric simplicial complex* is an abstract simplicial complex whose faces are materialized, and to which the following condition is added: the intersection of any two faces of the complex is either empty or a face of the complex.

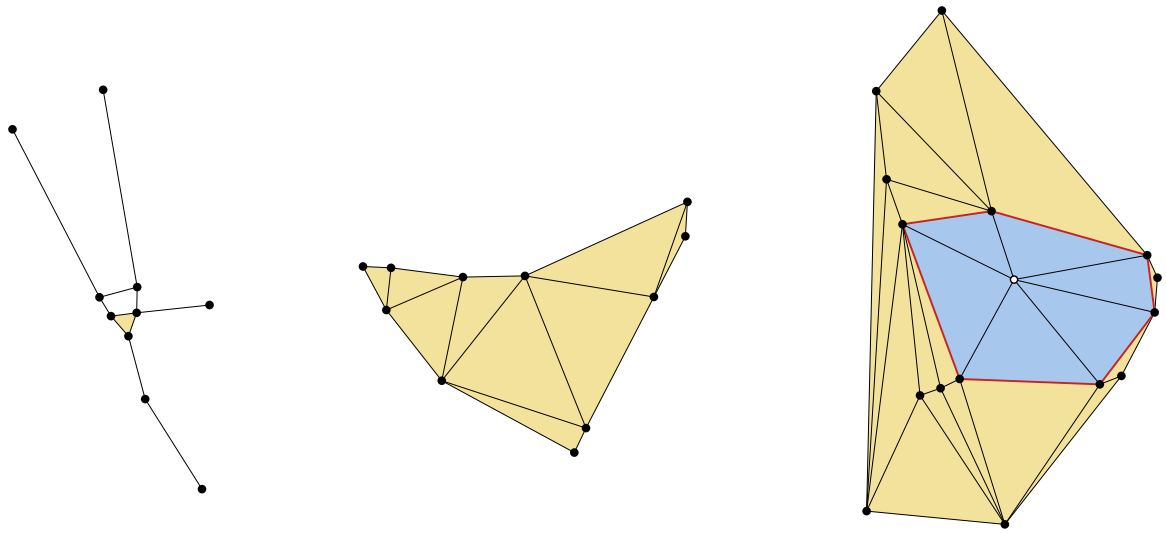

Figure 2.5: A simplicial complex (left) and a pure simplicial complex (center). On the right, the star of the white vertex is highlighted in blue, and its link in red.

2.6 Delaunay complex

The Delaunay complex has been widely studied and a few equivalent definitions are possible. We choose to stay in an abstract setting, in accordance with the way we introduced simplicial complexes.

A *Delaunay ball* is a maximal empty ball. Specifically, $B = B(x, r)$ is a Delaunay ball if any empty ball centered at x is contained in B. A simplex σ is a *Delaunay simplex* if there exists some Delaunay ball B such that the vertices of σ belong to $\partial B \cap \mathcal{P}$. The *Delaunay complex* is the set of Delaunay simplices, and is denoted by $\mathrm{Del}(\mathcal{P})$.

A *cosphericity* is the configuration where $n+2$ points lie on the same $(n+1)$-sphere. This degenerate configuration must be avoided to preserve the uniqueness of the Delaunay triangulation for a set of points. In the following, we always assume that the vertices in $\mathcal{P}$ are in *general position*, meaning that there is no cosphericity in the point set.

2.7 Duality of the Delaunay and Voronoi structures

The *nerve* of a finite covering C is defined as $\mathcal{N}(C) := \{D \subseteq C \mid \cap D \neq \emptyset\}$. We follow the definition of Edelsbrunner and Shah [69] and define the *realization* of the nerve as a simplicial complex $\mathcal{K}$ and a bijection φ between $\mathcal{N}(C)$ and $\mathcal{K}$ that preserves the inclusion, that is τ and σ are simplices of $\mathcal{K}$ such that $\tau \subset \sigma$ if, and only if, there exists $D_\sigma = \varphi(\sigma)$ and $D_\tau = \varphi(\tau)$ in $\mathcal{N}(C)$ such that $D_\sigma \subset D_\tau$. In the following, we identify $\mathcal{K}$ and the realization of the nerve.

The nerve of a Voronoi diagram $\mathrm{Vor}_d(\mathcal{P})$ is given by

$$\mathcal{N}(\mathrm{Vor}_d(\mathcal{P})) = \{\{\mathrm{V}_d(p_i)\} \mid \cap_i \mathrm{V}_d(p_i) \neq \emptyset, p_i \in \mathcal{P}\}.$$

Note that there is a complete combinatorial equivalence between $\mathcal{N}(\mathrm{Vor}_d(\mathcal{P}))$ and the Delaunay complex as defined in the previous section. A (partial) geometric realization of $\mathcal{N}(\mathrm{Vor}_d(\mathcal{P}))$ with vertices in $\mathcal{P}$ is called a *dual* of the Voronoi diagram.

If there exists an abstract homeomorphism between the nerve $\mathcal{N}(C)$ and the domain, then we say that the realization of the nerve is a *triangulation*.

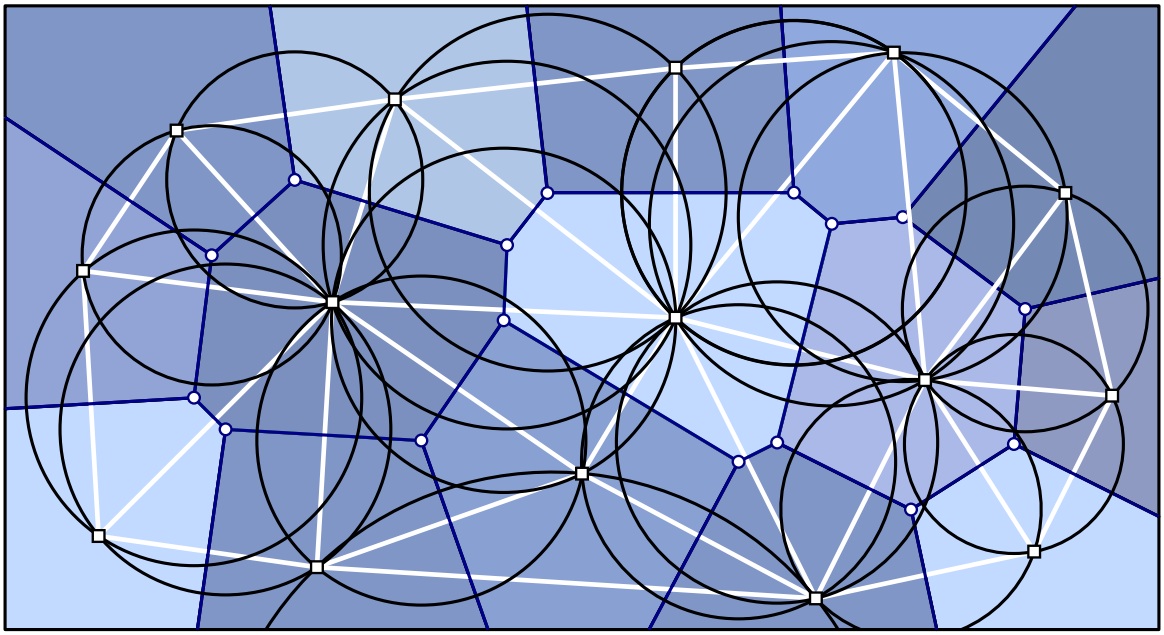

Figure 2.6: Delaunay triangulation (white) and Voronoi diagram (blue).

2.7.1 Dual of a simplex

The dual of a simplex is the set of centers of empty Delaunay balls. Equivalently, the dual of a k-simplex is a $(n-k)$-Voronoi face defined by the intersection of the $k+1$ Voronoi cells of the vertices of the simplex. For example in a 3D setting, the dual of a cell (a 3-simplex) is a Voronoi vertex (a 0-face), the dual of a facet (a triangle) is a Voronoi edge, the dual of an edge is a Voronoi 2-face and the dual of a point is a Voronoi cell.

2.8 Riemannian Voronoi diagram

Given a metric field g, the *Riemannian Voronoi diagram* of a point set $\mathcal{P}$, denoted by $\mathrm{Vor}_g(\mathcal{P})$, is the Voronoi diagram built using the geodesic distance d_g. Formally, it is a partition of the domain in *Riemannian Voronoi cells* $\{V_g(p_i)\}$, where

$$\mathrm{V}_g(p_i) = \{x \in \Omega \mid d_g(p_i, x) \leq d_g(p_j, x), \forall p_j \in \mathcal{P} \setminus p_i\}.$$

The Riemannian Voronoi diagram has received comparatively little attention as the complexity of computing geodesics makes it often undesirable to use in practice, especially when the manifold is endowed with an arbitrary metric field. Chapters III.6, III.7 and III.8 are devoted to the theoretical and practical study of this particular type of Voronoi diagram.

2.9 Riemannian Delaunay triangulations

In this thesis, we will consider both abstract simplices and complexes, as well as their geometric realization in $\mathbb{R}^n$ with vertex set $\mathcal{P}$. We now introduce two realizations of a simplex that will be useful, one curved and the other one straight.

The *straight realization* of a n-simplex σ with vertices in $\mathcal{P}$ is the convex hull of its vertices. We denote it by $\bar{\sigma}$. In other words,

$$\bar{\sigma} = \{x \in \Omega \subset \mathbb{R}^n \mid x = \sum_{p \in \sigma} \lambda_p(x)\, p, \lambda_p(x) \geq 0, \sum_{p \in \sigma} \lambda_p(x) = 1\}. \tag{2.3}$$

The *curved realization*, noted $\tilde{\sigma}$ is based on the notion of Riemannian center of mass [83, 65]. Let y be a point of $\bar{\sigma}$ with barycentric coordinate $\lambda_p(y), p \in \sigma$. We can associate the energy functional $\mathcal{E}_y(x) = \frac{1}{2}\sum_{p\in\sigma} \lambda_p(y) d_g(x,p)^2$. We then define the curved realization of σ as

$$\tilde{\sigma} = \{\tilde{x} \in \Omega \subset \mathbb{R}^n \mid \tilde{x} = \arg\min \mathcal{E}_{\bar{x}}(x), \bar{x} \in \bar{\sigma}\}. \tag{2.4}$$

The edges of $\tilde{\sigma}$ are geodesic arcs between the vertices. Such a curved realization is well defined provided that the vertices of σ lie in a sufficiently small ball according to the following theorem of Karcher [83].

Theorem 2.9.1 (Karcher) *Let the sectional curvatures K of Ω be bounded, that is $\Lambda_- \leq K \leq \Lambda_+$. Let us consider the function $\mathcal{E}_y$ on B_ρ, a geodesic ball of radius ρ that contains the set $\{p_i\}$. Assume that $\rho \in \mathbb{R}^+$ is less than half the injectivity radius and less than $\pi/4\sqrt{\Lambda_+}$ if $\Lambda_+ > 0$. Then $\mathcal{E}_y$ has a unique minimum point in B_ρ, which is called the* center of mass*.*

Given an (abstract) simplicial complex $\mathcal{K}$ with vertices in $\mathcal{P}$, we define the straight (resp., curved) realization of $\mathcal{K}$ as the collection of straight (resp., curved) realizations of its simplices, and we write $\bar{\mathcal{K}} = \{\bar{\sigma}, \sigma \in \mathcal{K}\}$ and $\tilde{\mathcal{K}} = \{\tilde{\sigma}, \sigma \in \mathcal{K}\}$.

We will consider the case where $\mathcal{K}$ is $\mathrm{Del}_g(\mathcal{P})$. A simplex of $\overline{\mathrm{Del}}_g(\mathcal{P})$ will simply be called a straight Riemannian Delaunay simplex and a simplex of $\widetilde{\mathrm{Del}}_g(\mathcal{P})$ will be called a curved Riemannian Delaunay simplex, omitting "realization of". In Chapters III.7 and III.8, we give sufficient conditions for $\overline{\mathrm{Del}}_g(\mathcal{P})$ and $\widetilde{\mathrm{Del}}_g(\mathcal{P})$ to be embedded in Ω, in which case we will call them the straight and the curved Riemannian Delaunay *triangulations* of $\mathcal{P}$.

Remark 2.9.2 *A curved Riemannian Delaunay n-simplex σ also satisfies the empty ball property: $\tilde{\sigma}$ is circumscribed by a geodesic ball that does not contain any point $q \in \mathcal{P}$ such that $q \notin \mathrm{Vert}(\sigma)$.*

2.10 Power diagram

For a chosen distance and seed set, other diagrams related to the Voronoi diagram can be constructed by enriching the definition of the cell of a seed p_i. For example, the *farthest Voronoi diagram* is defined through the cell

$$\mathrm{V}_d(p_i) = \{x \in \Omega \mid\ d(p_i, x) \geq d(p_j, x), \forall p_j \in \mathcal{P} \setminus p_i\},$$

meaning that a Voronoi cell is now the set of points that are *farther* from p_i than any other seed. Another possibility is to attribute weights to the seeds, either multiplicatively:

$$\mathrm{V}_d(p_i) = \{x \in \Omega \mid\ \omega_i d(p_i, x) \leq \omega_j d(p_j, x), \forall p_j \in \mathcal{P} \setminus p_i\},$$

or additively:

$$\mathrm{V}_d(p_i) = \{x \in \Omega \mid\ d(p_i, x) + \omega_i \leq d(p_j, x) + \omega_j, \forall p_j \in \mathcal{P} \setminus p_i\}.$$

When additive weights are used along with the squared Euclidean distance, one obtains the *additively weighted Voronoi diagram*, also known as the *power diagram* or the *Laguerre diagram* of a set of points, which we denote by $\mathrm{PD}_d(\mathcal{P})$. The *power cell* $\mathrm{V}_d^{\omega}(p_i)$ is given by

$$\mathrm{V}_d(p_i) = \{x \in \Omega \mid\ d(p_i, x)^2 + \omega_i \leq d(p_j, x)^2 + \omega_j, \forall p_j \in \mathcal{P} \setminus p_i\}.$$

Geometrical objects related to the power diagram are tagged with the *power* prefix: the distance is called the *power distance*, a face of the power diagram is called a *power face*, etc.

These new types of Voronoi diagrams can be considered using any distance – including the geodesic distance – but the study of these diagrams goes beyond the scope of this thesis and our use of power diagrams is limited to the Euclidean setting.

Sectional Voronoi diagram The intersection of an Euclidean Voronoi diagram $\mathrm{Vor}(\mathcal{P})$ living in an ambient space Ω with an affine space $\mathcal{A} \subset \Omega$ is a lower-dimensional power diagram $\mathrm{PD}(\mathcal{P}')$, where $\mathcal{P}'$ is obtained by projecting orthogonally $\mathcal{P}$ onto $\mathcal{A}$. Indeed, letting $\Pi_{\mathcal{A}}$ be the orthogonal projection onto $\mathcal{A}$ and taking $p_i \in \mathcal{P}$, we have

$$\forall x \in \mathcal{A}, \|x - p_i\|^2 = \|x - \Pi_{\mathcal{A}}(p_i)\|^2 + \|\Pi_{\mathcal{A}}(p_i) - p_i\|^2 .$$

In the ambient space, the Voronoi cell of p_i is the set of points $x \in \Omega$ such that

$$\|x - \Pi_{\mathcal{A}}(p_i)\|^2 + \|\Pi_{\mathcal{A}}(p_i) - p_i\|^2 \leq \|x - \Pi_{\mathcal{A}}(p_j)\|^2 + \|\Pi_{\mathcal{A}}(p_j) - p_j\|^2 ,$$

for $p_j \in \mathcal{P}$ with $p_j \neq p_i$. This inequality can be equivalently seen as the definition of the power cell of the point $\Pi_{\mathcal{A}}(p_i)$ with associated weight $\omega_i = -\|\Pi_{\mathcal{A}}(p_i) - p_i\|^2$ in the affine space $\mathcal{A}$.

2.10.1 Regular Delaunay triangulation

The dual of a power diagram is a triangulation, called the *regular Delaunay triangulation*. As the power diagram generalizes the Voronoi diagram, the regular Delaunay triangulation generalizes the Delaunay triangulation.

2.11 Restricted Delaunay triangulation

The Delaunay and Voronoi structures presented so far are built from (almost) arbitrary point sets living in $\mathbb{R}^n$. It is possible to employ these structures to approximate bounded domains. The *restriction* of a Delaunay complex $\mathrm{Del}(\mathcal{P})$ to a domain Ω is the subcomplex (the *restricted Delaunay complex*) $\mathrm{Del}_{|\Omega}(\mathcal{P})$ composed of the simplices of $\mathrm{Del}(\mathcal{P})$ whose dual Voronoi face intersects Ω.

Restricted Delaunay triangulations were introduced by Chew [53] and allow to accurately capture complex geometric objects. For example, it can be shown that under the condition of good sampling of a surface, the restricted Delaunay triangulation and the domain are homeomorphic [9]. Thanks to these good properties, restricted Delaunay triangulations have often been used to create provably correct refinement algorithms in the case of surfaces [45, 27, 120]. It was however proven by Boissonnat, Guibas and Oudot [26] that this does not extended to higher-dimensional settings. Nevertheless, restricted Delaunay triangulations will be consistently used in the refinement algorithms considered and presented in this thesis.

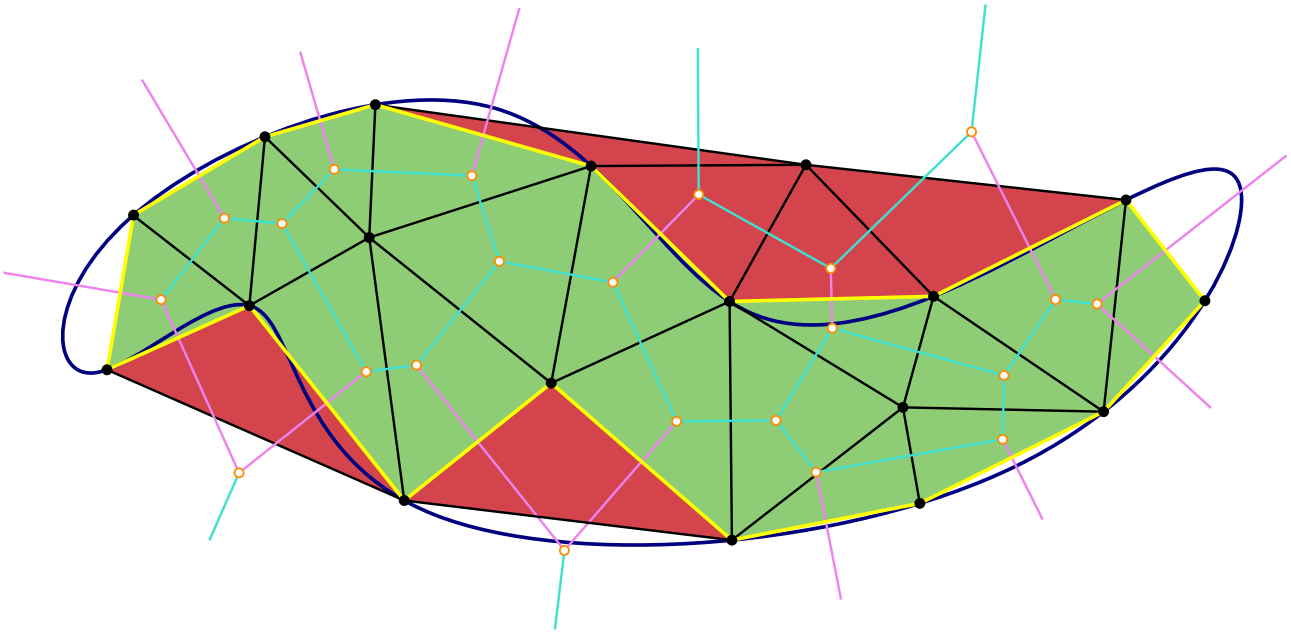

Figure 2.7: a 2D Delaunay triangulation of a set of vertices (black) restricted to a curve (blue). Voronoi edges are represented in teal, and in pink if they intersect the curve. The Voronoi vertices are marked with orange circles. Restricted Delaunay edges are drawn in yellow, and restricted Delaunay triangles are drawn in green.

2.12 Assumptions on point sets

Studying and inferring properties on geometric structures is difficult when using completely random point sets. It is thus natural to assume hypotheses on the point set: for example, one of the most basic assumptions made when building Delaunay complexes is that points are in general position (see Section 2.6). We introduce two different notions to control the quality of the samplings that we consider and of the elements that we generate with such point sets.

Definitions are given here in a Riemannian setting (meaning, with respect to the geodesic distance), but can naturally be applied to simpler distances: the isotropic distance in $\mathbb{R}^n$ can for example be seen as a Riemannian manifold endowed with a Euclidean metric field.

2.12.1 Nets

Controlling the density and the sparsity of a point set allows to deduce properties on the shape of simplices of the complexes built using the point set. The idea of imposing such constraints on point sets is not recent and appears in various domains: Delone [1] sets, for example, employ a *packing radius* to ensure that every point of the domain is close to a sample point and are used in coding theory and crystallography. More closely to our topic, nets have notably found use in surface reconstruction [8, 25] where they are used to create "good" (with respect to the algorithm) samplings of the domain. The concept of nets presented here conveys these requirements through *sampling* and *separation* parameters.

[1] Delone and Delaunay are in fact the same person.

Sampling The sampling parameter is used to control the density of a point set: if Ω is a bounded domain, $\mathcal{P}$ is said to be an *ε-sample set* for Ω with respect to a metric field g if $d_g(x, \mathcal{P}) < \varepsilon$ for all $x \in \Omega$.

Separation The sparsity of a point set is controlled by the separation parameter: the set $\mathcal{P}$ is said to be μ-separated with respect to a metric field g if $d_g(p, q) \geq \mu$ for all $p, q \in \mathcal{P}$. We assume that the separation parameter μ is proportional to the sampling parameter ε and thus there exists a positive λ, with $\lambda \leq 1$, such that $\mu = \lambda\varepsilon$.

If $\mathcal{P}$ is an ε-sample set that is μ-separated, we say that $\mathcal{P}$ is an *(ε, μ)-net*. Figure 2.8 shows an Euclidean net with $\lambda = 1/3$.

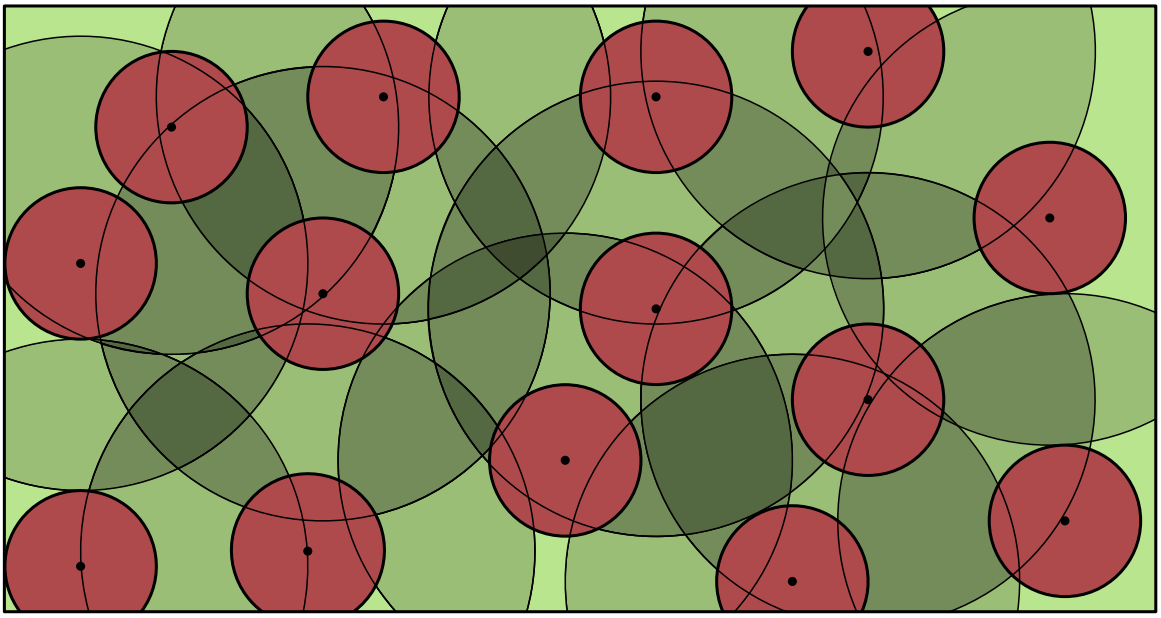

Figure 2.8: An Euclidean net with $\varepsilon/\mu = 3$. The red circles represent the separation and the green circles the density of the sample.

Remark 2.12.1 *Every point set is a net for a sufficiently small μ and a sufficiently large ϵ, but the quality of the result is generally increased for stricter conditions.*

Building Delaunay triangulations and Voronoi diagrams using nets guarantee properties on the quality of their elements; those properties will be detailed in Chapter 3.

2.12.2 Protection and power protection of point sets

Protection of simplices is a concept formally introduced by Boissonnat, Dyer and Ghosh [17], which targets Delaunay triangulations. The main idea is that no vertex should lie close to the circumscribing balls of the Delaunay simplices of a triangulation. A similar scheme was first introduced by Funke et al. [74], who perturbed vertices of a Delaunay triangulation away from the Delaunay balls of other simplices to ensure arithmetical robustness of the triangulation. The concept of protection was extended by Boissonnat et al. [22].

We now define the protection of a simplex, and of a point set. Let σ be a Delaunay simplex whose vertices belong to $\mathcal{P}$, and whose (empty) Delaunay ball is noted $B(c, r)$, where c is the circumcenter

and r is the circumradius. For $0 \leq \delta \leq r$, we associate to $B(c,r)$ the dilated balls $B^{+\delta} = B(c, r+\delta)$ and $B_p^{+\delta} = B(c, \sqrt{r^2+\delta^2})$. We say that σ is *δ-protected* (resp. *δ-power protected*) if $B^{+\delta}$ (resp. $B_p^{+\delta}$) is empty of any vertex in $\mathcal{P} \setminus \mathrm{Vert}(\sigma)$. The balls $B^{+\delta}$ (resp. $B_p^{+\delta}$) is the *power* (resp. *power-protected*) ball of σ. If all the simplices of $\mathrm{Del}(\mathcal{P})$ are δ-protected (resp. δ-power protected) then $\mathcal{P}$ is δ-protected (resp. δ-power protected). We assume that the protection parameter δ is proportional to the sampling parameter ε, thus there exists a positive ι, with $\iota \leq 1$, such that $\delta = \iota\varepsilon$. We will prefer power protection to protection, as we often compare squared distances. Figure 2.9 shows a protected point set with the protection zones.

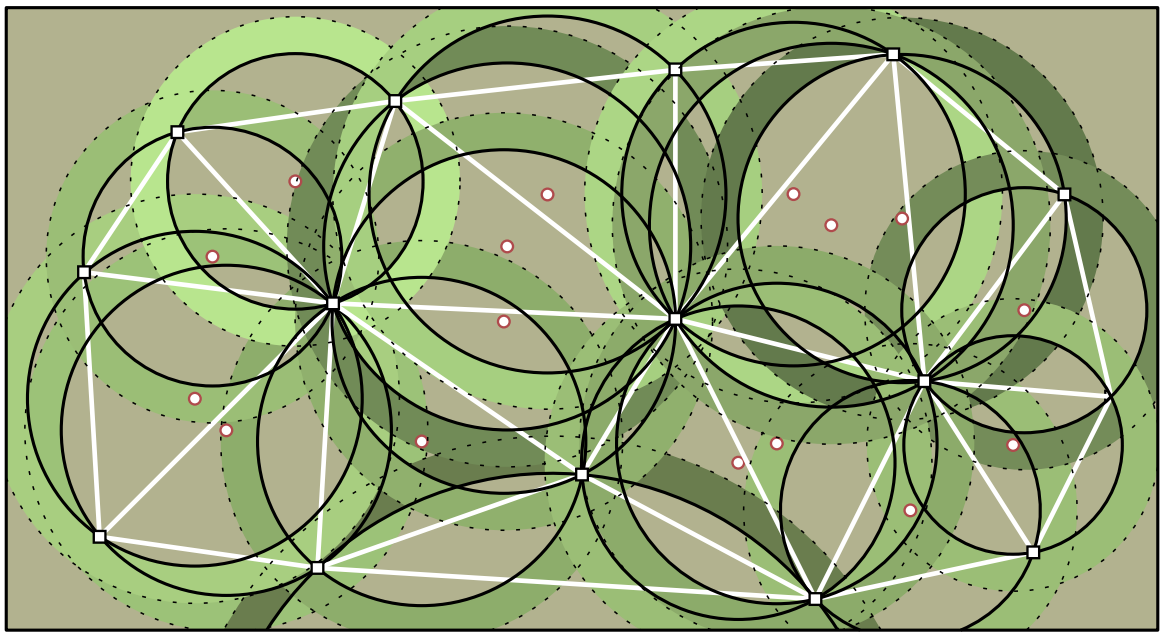

Figure 2.9: A protected point set. The (Euclidean) Delaunay triangulation is drawn in white. The circumcenters are red circles. For each Delaunay simplex, the Delaunay ball is the black circle, the green zone the protection zone and the dotted circle the protected Delaunay ball.

Adopting the point of view of the Voronoi diagram, we can define protection and power protection as follows. If v, a Voronoi vertex, is the intersection of $n+1$ Voronoi cells $\{\mathrm{V}_d(p_i)\}$ with $\{p_i\} := \mathcal{P}' \subset \mathcal{P}$, then the simplex dual of v is

- protected, if $d(v,p_j) > d(v,p_i) + \delta$
- power protected, if $d(v,p_j)^2 > d(v,p_i)^2 + \delta^2$

for $p_j \notin \mathcal{P}'$. This point of view is also useful when the use of balls is not convenient or adapted to the distance considered.

Remark 2.12.2 *Similarly to nets, every point set that is not degenerate is protected (or power protected) for a small enough δ.*

Assuming that a point set is protected is specifically useful to deduce bounds on the separation of non-adjacent Voronoi entities (and in particular of Voronoi vertices), that is the existence of a lower bound on the distance between two non-adjacent Voronoi entities. These properties are detailed in Chapter 3 and will be essential to prove the theoretical correctness of both the anisotropic Voronoi diagram defined by Labelle and Shewchuk in Chapter II.5 and of our discrete Riemannian Voronoi diagrams in Chapters III.7 and III.8.

Relating protection and power protection

The concepts of protection and power protection are closely related. Indeed, if we assume δ_p-protection, we have by definition that

$$d(c,q) \geq r + \delta_p,$$

and

$$d(c,q)^2 \geq r^2 + \delta_p^2 + 2r\delta,$$

giving δ_{pp}-power protection with $\delta_{pp} = \sqrt{\delta^2 + 2r\delta} = \delta\sqrt{1 + \frac{2r}{\delta}}$.

On the other hand, if we assume δ_{pp}-power protection, we have that

$$d(c,q)^2 \geq r^2 + \delta_{pp}^2.$$

Thus

$$d(c,q) \geq \sqrt{r^2 + \delta_{pp}^2} = r\sqrt{1 + \frac{\delta_{pp}^2}{r^2}} = r\left(1 + \frac{1}{2}\frac{\delta_{pp}^2}{r^2} + o(r^2)\right) = r + \frac{\delta_{pp}^2}{2r} + o(r),$$

and $\delta_p \sim \frac{\delta_{pp}^2}{2r}$.

Whether protection or power protection is thus only linked to the other objects being considered. If squared distances are being compared, it is simpler to work with power protection.

2.13 Unit mesh

In the case of Delaunay refinement algorithms, the generation of meshes is governed by a set of criteria, arbitrarily-chosen rules that all simplices must satisfy for the refinement to finish. Refinement algorithm traditionally employ a *sizing* criterion to control the maximal size of simplices, by imposing an upper bound on either the maximal lengths of any edge or the circumradius of a simplex. While this upper bound can be constant over the domain, elements of varying sizes are generally desired (similarly to how a single metric is not sufficient to describe the anisotropy over the domain). A *sizing field* is naturally defined to provide a scalar map which associates to any point of a domain, the maximal allowed length of edges (or circumradius). For example, Ruppert [122] defines a size function that depends on the distance to the boundary.

Remark 2.13.1 *If the sizing criterion applies a maximal value on the length of the circumradius, it is nevertheless possible to compute a bound on the length of edges of the same simplex since the length of an edge of a triangle in a Delaunay triangulation is bounded by twice the circumradius of the triangle.*

When the edges of a triangulation have length 1, the mesh is said to be *unit* [72]. Naturally, a mesh whose edges are unit is difficult (if it is even physically possible) to achieve. A tolerance $\varsigma > 1$ is generally introduced to allow the construction of almost-unit meshes whose edges have a length between $1/\varsigma$ and ς.

In the context of anisotropic meshes, the lengths of edges are computed in the metric and the metric thus naturally encodes a sizing information through its eigenvalues. A sizing field is thus already incorporated within the metric field. A change in the scale of the sizing field can be simply achieved by scaling the eigenvalues.

2.13.1 Unit meshes and distortion

When computing edge lengths in the metric of the edge's endpoints, the distortion between adjacent vertices of a mesh can be bounded when that mesh is unit. Indeed, we know that the distortion $\psi(p, q)$ between two points p and q can be used to relate the distance in the metrics G_p and G_q (Equation 2.1). Thus if we allow a tolerance ς on the unit mesh, the maximal distortion that is possible in a unit mesh is ς^2. In various mesh generators, the tolerance is set to $\sqrt{2}$, and thus the distortion between two adjacent vertices cannot be greater than 2, or the mesh is not unit.

When creating metric fields, the sizing field is always reasonably chosen (small enough) such that a unit mesh is physically possible.

3. Preliminary results

This chapter details properties of the basic notions introduced in Chapter 2 and results inferred using these notions. As it mainly consists of technical lemmas that are stepping stones towards central results in Chapters II.5, III.7 and III.8, it can be skipped on first reading and simply consulted as needed.

The three important notions upon which a majority of our work rests are the notions of distortion, net and power protection. The first part of this chapter is dedicated to a short study of the distortion. We first look into some properties and shortcomings of the classical definition of Labelle and Shewchuk, and introduce an alternate definition for the distortion to remedy these disadvantages. The subsequent sections present properties that are inferred assuming that the point set is a power protected net. We study the Euclidean Voronoi diagram and the Euclidean Delaunay triangulation when built from power protected nets. We expose lower bounds on the distance between non-adjacent Voronoi faces and lower and upper bounds on the dihedral angles between adjacent faces of these structures. The combination of these results allows to deduce the stability of the power protected net assumption when metrics and metric fields are perturbed.

Contributions All the results detailed in this chapter are new. We explain how to compute the sampling parameter of a sample such that the point set satisfies an upper bound on the distortion between adjacent vertices (for the original definition given by Labelle and Shewchuk). We propose an alternate definition for the distortion that possesses the same properties as the original definition, but is more meaningful geometrically and does not require strong assumptions to be related to the sampling of the points.

Some lemmas of the second part of this chapter are close to known results or aim to prove similar properties, for example the stability of the power protection hypothesis with respect to metric perturbation. However, both the assumptions – on the (power) protection and the distortion – as well as the path followed to achieve the result differ sufficiently for us to consider these results new. We nevertheless indicate when it is the case, and where the previous result has been introduced.

Contents

3.1 Distortion

In previous works on anisotropic mesh generation [89, 50, 38] as well as in the study of our own algorithms, the good behavior of algorithms, for example the embedding of a triangulation, is generally guaranteed by upper bounds on the density of the point set and on the maximal distortion between adjacent vertices. A bound on the distortion is needed as it is difficult to handle widely changing metric fields, similarly to how it is difficult to handle manifolds with poor sampling.

A requirement on the maximal distortion between adjacent vertices being somewhat abstract, we seek to express a bound on the distortion as a requirement on the sampling of the point set. We first achieve this correspondence for the distortion introduced by Labelle and Shewchuk, but at the cost of unreasonable assumptions on the metric field. Consequently, we introduce a new definition of distortion between two metrics.

Locality of the Labelle-Shewchuk distortion

Given a bound on the maximal distortion in a region, we wish to compute the maximal distance between two points of the sample such that their distortion satisfies the bound. Such a link can be exposed if we assume that the metric field is bi-Lipschitz. Indeed, Assume that there exists $\zeta \in \mathbb{R}$ such that

$$\frac{1}{\zeta} \|p - q\| \leq \|g(p) - g(q)\| \leq \zeta \|p - q\|, \forall p, q \in \Omega.$$

Lemma 3.1.1 gives the relation between the size of a neighborhood and the maximal distortion between two points of that neighborhood.

Lemma 3.1.1 *Let g be a ζ-bi-Lipschitz Riemannian metric field. Let p_0 be a point of Ω and $\psi_0 \geq 1$ a distortion bound. If $\psi(g(p), g(p_0)) \leq \psi_0$, then*

$$\left\|p - p_0\right\| \leq \zeta \left(1 + \psi_0^2\right) \left\|g(p_0)\right\|.$$

Proof. We have

$$\begin{aligned}
\left\|g(p) - g(p_0)\right\| &= \left\|g(p)^{\frac{1}{2}} g(p_0)^{-\frac{1}{2}} g(p_0) g(p_0)^{-\frac{1}{2}} g(p)^{\frac{1}{2}} - g(p_0)\right\| \\
&\leq \left\|g(p_0)\right\| + \left\|g(p)^{\frac{1}{2}} g(p_0)^{-\frac{1}{2}} g(p_0) g(p_0)^{-\frac{1}{2}} g(p)^{\frac{1}{2}}\right\| \\
&\leq \left\|g(p_0)\right\| + \left\|g(p)^{\frac{1}{2}} g(p_0)^{-\frac{1}{2}}\right\| \left\|g(p_0)\right\| \left\|g(p_0)^{-\frac{1}{2}} g(p)^{\frac{1}{2}}\right\| \\
&\leq \left\|g(p_0)\right\| \left(1 + \psi_0^2\right)
\end{aligned}$$

Since the metric field is ζ-bi-Lipschitz, we have

$$\frac{1}{\zeta} \|p - p_0\| \leq \|g(p) - g(p_0)\|,$$

giving us a condition on $\|p - p_0\|$:

$$\|p - p_0\| \leq \zeta(1 + \psi_0^2) \|g(p_0)\|.$$

□

A bi-Lipschitz requirement is however unrealistic as most metric fields are in fact not bi-Lipschitz: uniform metric fields are, for example, not bi-Lipschitz.

A new definition of the distortion

As we cannot reasonably assume that the metric field is bi-Lipschitz, we introduce a new definition of distortion. A prerequisite to our definition is that the metric field is (only) ζ-Lipschitz, meaning that $\|G_p - G_q\| \leq \zeta \, |p - q|$ with $\zeta \in \mathbb{R}^+$. Contrary to the bi-Lipschitz assumption that was previously needed, this is perfectly reasonable. Our distortion is given by

$$\psi(p,q) = \psi(G_p, G_q) = \max \left\{ \frac{1}{\left(1 - \sqrt{\frac{2\zeta r_p \sqrt{\lambda_{p,\max}}}{\lambda_{p,\min}}}\right)}, \, 1 + \sqrt{\frac{2\zeta r_p \sqrt{\lambda_{p,\max}}}{\lambda_{p,\min}}} \right\}, \tag{3.1}$$

where r_p is a bound on the distance between p and q measured with respect to G_p, and $\lambda_{p,\min}$ and $\lambda_{p,\max}$ are respectively the smallest and largest eigenvalues of G_p.

A first observation is that we have $\psi(G_p, G_q) \geq 1$ for two metrics G_p and G_q. However, there is a small difference for the limit case of $\psi = 1$. Indeed, we do not have that $G_p = G_q$ implies $\psi(G_p, G_q) = 1$ as Labelle and Shewchuk do. Nevertheless, the distortion goes to 1 with equality in the limit as the neighborhood becomes smaller (*i.e.* the bound r_p goes to 0).

That our definition does not automatically attribute a small distortion to metrics that are far (even if they are close) is not an issue and might even be seen as a beneficial trait: a small distortion with the definition of Labelle and Shewchuk can sometimes be misleading, *e.g.* by concluding that the metric field is relatively constant between two distant points because their distortion is small, even though important variations may exist in between. Our distortion expresses more clearly the convergence of the distortion between two points as they become closer.

We prove in Lemma 3.1.2 that our definition also allows to relate distances in different metrics.

Lemma 3.1.2 *Let U be a neighborhood centered on p and of radius r_p with respect to $G_p = g(p)$. Let $q \in U$. The distances d_p and d_q can be related by*

$$\frac{1}{\psi(p,q)} \, d_p(x,y) \leq d_q(x,y) \leq \psi(p,q) \, d_p(x,y)$$

with $\psi(p,q)$ defined as in Equation 3.1.

Proof. Let $G_q = G_p + E$. We have

$$\dot{\gamma}^t (G_q - G_p) \dot{\gamma} = \dot{\gamma}^t E \dot{\gamma}$$

Thus

$$\left| \dot{\gamma}^t (G_q - G_p) \dot{\gamma} \right| \leq \left| \dot{\gamma}^t E \dot{\gamma} \right| = \dot{\gamma} \cdot (E\dot{\gamma}) \leq |\dot{\gamma}| \, |E\dot{\gamma}| \leq \|E\| \, |\dot{\gamma}|^2 = \|E\| \, \dot{\gamma} \cdot \dot{\gamma}.$$

Because metrics are given by symmetric matrices, the eigenvalues of G_p can be used to obtain bounds:

$$\lambda_{p,\min} \, \dot{\gamma} \cdot \dot{\gamma} \leq \dot{\gamma}^t G_p \dot{\gamma} \leq \lambda_{p,\max} \, \dot{\gamma} \cdot \dot{\gamma}, \tag{3.2}$$

where $\lambda_{p,\min}$ and $\lambda_{p,\max}$ are the smallest and largest eigenvalues of G_p respectively. We see that

$$\begin{aligned}
|\ell_{G_q}(\gamma) - \ell_{G_p}(\gamma)| &= \left| \int \sqrt{\dot\gamma^t G_q \dot\gamma}\, \mathrm{d}t - \int \sqrt{\dot\gamma^t G_p \dot\gamma}\, \mathrm{d}t \right| \\
&= \left| \int \sqrt{\dot\gamma^t (G_q - G_p) \dot\gamma}\, \mathrm{d}t \right| \\
&\leq \sqrt{\|E\|} \left| \int \sqrt{\dot\gamma \cdot \dot\gamma}\, \mathrm{d}t \right| \\
&\leq \sqrt{\|E\|} \left| \int \sqrt{\frac{\dot\gamma^t G_p \dot\gamma}{\lambda_{p,\min}}}\, \mathrm{d}t \right| \\
&= \sqrt{\frac{\|E\|}{\lambda_{p,\min}}}\, \ell_{G_p}(\gamma),
\end{aligned}$$

where $\ell_{G_q}(\gamma)$ and $\ell_{G_p}(\gamma)$ denote the lengths of γ with respect to G_q and G_p respectively. This means that

$$\left(1 - \sqrt{\frac{\|E\|}{\lambda_{p,\min}}}\right) \ell_{G_p}(\gamma) \leq \ell_{G_q}(\gamma) \leq \left(1 + \sqrt{\frac{\|E\|}{\lambda_{p,\min}}}\right) \ell_{G_p}(\gamma),$$

Let us focus on the second inequality, and let $\gamma_{(x,y),G_p}$ denotes the shortest path with respect to G_p connecting x and y. We find

$$d_g(x,y) \leq \ell_{G_q}\left(\gamma_{(x,y),G_p}\right) \leq \left(1 + \sqrt{\frac{\|E\|}{\lambda_{p,\min}}}\right) \ell_{G_p}\left(\gamma_{(x,y),G_p}\right) = \left(1 + \sqrt{\frac{\|E\|}{\lambda_{p,\min}}}\right) d_{G_p}(x,y).$$

A similar statement holds for the first inequality:

$$\left(1 - \sqrt{\frac{\|E\|}{\lambda_{p,\min}}}\right) d_{G_p}(x,y) \leq d_g(x,y) \leq \left(1 + \sqrt{\frac{\|E\|}{\lambda_{p,\min}}}\right) d_{G_p}(x,y).$$

We assume that any metric field that we consider is ζ-Lipschitz, that is

$$\|E\| = \|G_p - G_q\| \leq \zeta\, |p - q|\,.$$

Note that, because of (3.2) and using similar reasoning as before, we have

$$\sqrt{\lambda_{p,\min}}\, d_{G_p}(p,q) \leq |p - q| \leq \sqrt{\lambda_{p,\max}}\, d_{G_p}(p,q).$$

Which means that

$$\left(1 - \sqrt{\frac{2\zeta r_p \sqrt{\lambda_{p,\max}}}{\lambda_{p,\min}}}\right) d_{G_p}(x,y) \leq d_g(x,y) \leq \left(1 + \sqrt{\frac{2\zeta r_p \sqrt{\lambda_{p,\max}}}{\lambda_{p,\min}}}\right) d_{G_p}(x,y),$$

where r_p is a bound on the size of the neighborhood. □

Note that our distortion was developed so that both distortions satisfy the same equations, which means that when this property is used, it does not matter which definition of the distortion is used.

Remark 3.1.3 *We have intentionally used geodesic distance expressions (that is, considering $\int \gamma_{G_p}(t)$ and not simply d_{G_p}) despite only considering metrics. This can be seen as abusively considering the two metrics as uniform metric fields over the domain. Additionally, while the relation that was proven concerned d_{G_p} and d_{G_q}, we did not make use of the constancy of G_q in the proof. Therefore, we could actually reproduce the proof for a non-uniform metric field g instead of G_q, therefore linking the "uniform" distance d_p and the geodesic distance d_g.*

All the core properties of the distortion proposed by Labelle and Shewchuk are thus also present in our definition.

Although slightly more demanding on the metric field and more technical, our definition only requires the metric field to be Lipschitz continuous (and not bi-Lipschitz, as was needed in Section 3.1), follows a more natural geometrical approach, and offers a clear link between sampling and distortion, as proven in the next section.

Locality with our new distortion

For a given distortion bound, we obtain immediately a local upper bound sought on the distance between points. Indeed, if $\psi(p,q) \leq \psi_0$, we must have

$$\frac{1}{\left(1-\sqrt{\frac{2\zeta r_p\sqrt{\lambda_{p,\max}}}{\lambda_{p,\min}}}\right)} \leq \psi_0 \quad \text{and} \quad 1+\sqrt{\frac{2\zeta r_p\sqrt{\lambda_{p,\max}}}{\lambda_{p,\min}}} \leq \psi_0.$$

Equivalently,

$$r_p \leq \left(1-\frac{1}{\psi_0}\right)^2 \frac{\lambda_{p,\min}}{2\zeta\sqrt{\lambda_{p,\max}}} \quad \text{and} \quad r_p \leq (1+\psi_0)^2 \frac{\lambda_{p,\min}}{2\zeta\sqrt{\lambda_{p,\max}}}.$$

Both conditions can be merged, giving

$$r_p \leq \min\left\{\left(1-\frac{1}{\psi_0}\right)^2 \frac{\lambda_{p,\min}}{2\zeta\sqrt{\lambda_{p,\max}}}, (1+\psi_0)^2 \frac{\lambda_{p,\min}}{2\zeta\sqrt{\lambda_{p,\max}}}\right\}. \tag{3.3}$$

Meshes that we consider in this thesis are generated with variations of the classical Delaunay refinement algorithm. There is generally a size constraint that is imposed on elements, through an upper bound on the circumradius of simplices. The bound in Equation 3.3 can be converted into a sizing field constraint by noting that if circumradii are bounded by ℓ, the maximal length of an edge in the complex is bounded by 2ℓ. Thus, if p and q were two adjacent vertices in a triangulation for which we must have $\psi(p,q) \leq \psi_0$, a sizing bound on the sample would be given by $\varepsilon \leq r_p/2$.

In practice We give here an example of the way this relation is used in practice. In the proofs of the good behavior of our anisotropic Voronoi diagram, we will generally consider a neighborhood U around a point p_0 of an ε-sample $\mathcal{P}$ and require U to contain at least the Voronoi cell $\mathrm{V}(p)$ and the Voronoi cells of the neighbors of p_0. To achieve results, we will approximate the anisotropic Voronoi cell with a simpler cell, which is only reasonable if the distortion within U is below a given bound ψ_0. Assuming that the metric field is Lipschitz, we can then translate the distortion bound ψ_0 to a sampling bound on ε, the sampling parameter of $\mathcal{P}$. Indeed, we know that the ball $B_g(p_0, r_0)$ with $r_0 >= 5\varepsilon$ contains $\mathrm{V}(p_0)$ and its adjacent Voronoi cells; therefore, we must have $\varepsilon \leq r_0/5$. The value of r_0 is obtained from ψ_0 by Equation 3.3.

3.1.1 Geodesic distortion

The concept of distortion was originally introduced to relate distances with respect to two metrics, but, as claimed in Section 2.2.1, this result can be (locally) extended to geodesic distances.

Lemma 3.1.4 *Let $U \subset \Omega$ be open, and g and g' be two Riemannian metric fields on U. Let $\psi_0 \geq 1$ be a bound on the metric distortion. Suppose that U is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$, such that $\forall p \in B(p_0, r_0), \psi(g(p), g'(p)) \leq \psi_0$. Then, for all $x, y \in U$,*

$$\frac{1}{\psi_0} d_g(x,y) \leq d_{g'}(x,y) \leq \psi_0 \, d_g(x,y),$$

where d_g and $d_{g'}$ indicate the geodesic distances with respect to g and g' respectively.

Proof. Recall that for $p \in B_g(p_0, r_0)$ and, for any pair x, y of points, we have

$$\frac{1}{\psi_0} \, d_{g(p)}(x,y) \leq d_{g'(p)}(x,y) \leq \psi_0 \, d_{g(p)}(x,y).$$

Therefore, for any curve $\gamma(t)$ in U, we have that

$$\frac{1}{\psi_0} \int \sqrt{\langle \dot{\gamma}, \dot{\gamma} \rangle}_{g(\gamma(t))} \mathrm{d}t \leq \int \sqrt{\langle \dot{\gamma}, \dot{\gamma} \rangle}_{g'(\gamma(t))} \mathrm{d}t \leq \psi_0 \int \sqrt{\langle \dot{\gamma}, \dot{\gamma} \rangle}_{g(\gamma(t))} \mathrm{d}t.$$

Considering the infimum over all paths γ that begin at x and end at y, we find that

$$\frac{1}{\psi_0} d_g(x,y) \leq d_{g'}(x,y) \leq \psi_0 d_g(x,y).$$

□

Note that this result is independent from the definition of the distortion and is entirely based on the inequality comparing distances in two metrics (Equation 2.1), which was shown to be fulfilled for both definitions of the distortion (see Section 3.1.2).

3.2 Separation of Voronoi entities

Power protected point sets were introduced to create quality bounds for the simplices of Delaunay triangulations built using such point sets [17]. We will show that power protection allows to deduce additional useful results for Voronoi diagrams. In this section we show that when a Voronoi diagram is built using a power protected sample set, its non-adjacent Voronoi faces, and specifically its Voronoi vertices are separated. This result is essential to our proofs in Chapters II.5, III.7 and III.8 where we approximate complicated Voronoi cells with simpler Voronoi cells without creating inversions in the dual, which is only possible because we know that Voronoi vertices are far from one another.

3.2.1 Separation of Voronoi vertices

The main result provides a bound on the Euclidean distance between Voronoi vertices of the Euclidean Voronoi diagram of a point set and is given by Lemma 3.2.4. The following three lemmas are intermediary results needed to prove Lemma 3.2.4.

Lemma 3.2.1 *Let $B = B(c, R)$ and $B' = B(c', R')$ be two n-balls whose bounding spheres ∂B and $\partial B'$ intersect, and let H be the bisecting hyperplane of B and B',* i.e. *the hyperplane that contains the $(n-2)$-sphere $S = \partial B \cap \partial B'$. Let θ be the angle of the cone (c, S). Writing $\rho = \frac{R'}{R}$ and $\|c - c'\| = \lambda R$, we have*

$$\cos(\theta) = \frac{1 + \lambda^2 - \rho^2}{2\lambda}. \tag{3.4}$$

If $R \geq R'$, we have $\cos(\theta) \geq \frac{\lambda}{2}$.

Proof. Let $q \in S$; applying the cosine rule to the triangle $\triangle cc'q$ gives

$$\lambda^2 R^2 + R^2 - 2\lambda R^2 \cos(\theta) = R'^2, \tag{3.5}$$

which proves Equation (3.4). If $R \geq R'$, then $\rho \leq 1$, and $\cos(\theta) \geq \lambda/2$ immediately follows from Equation (3.4). □

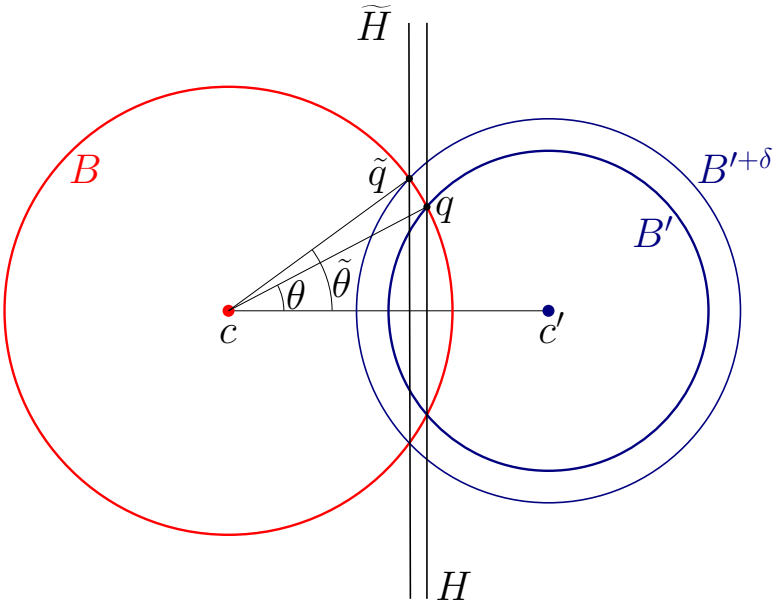

Figure 3.1: Construction used in Lemmas 3.2.1 and 3.2.2.

Lemma 3.2.2 *Let $B = B(c, R)$ and $B' = B(c', R')$ be two n-balls whose bounding spheres ∂B and $\partial B'$ intersect, and let $\tilde{\theta}$ be the angle of the cone $(c, \widetilde{S})$ where $\widetilde{S} = \partial B \cap \partial B'^{+\delta}$. Writing $\|c - c'\| = \lambda R$, we have*

$$\cos(\tilde{\theta}) = \cos(\theta) - \frac{\delta^2}{2R^2\lambda}$$

Proof. Let $\tilde{q} \in \tilde{S}$, applying the cosine rule to the triangle $\triangle cc'\tilde{q}$ gives

$$\lambda^2 R^2 + R^2 - 2\lambda R^2 \cos(\tilde{\theta}) = R'^2 + \delta^2.$$

Subtracting (3.5) from the previous equality yields $\delta^2 = 2\lambda R^2(\cos(\theta) - \cos(\tilde{\theta}))$, which proves the lemma. □

Lemma 3.2.3 *Let $\sigma = p * \tau$ and $\sigma' = p' * \tau$ be two Delaunay simplices sharing a common facet τ. (Here $*$ denotes the join operator.) Let $B(\sigma) = B(c, R)$ and $B(\sigma') = B(c', R')$ be the circumscribing balls of σ and σ' respectively. Then σ' is δ-power protected with respect to p, that is $p \notin B(\sigma')^{+\delta}$ if and only if $\|c - c'\| \geq \frac{\delta^2}{2D(p,\sigma)}$.*

Proof. The spheres ∂B and $\partial B'^{+\delta}$ intersect in a $(n-2)$-sphere $\widetilde{S}$ which is contained in a hyperplane $\widetilde{H}$ parallel to the hyperplane $H = \mathrm{Aff}(\tau)$. For any $\tilde{q} \in \widetilde{S}$ we have

$$d(\widetilde{H}, H) = d(\tilde{q}, H) = R(\cos(\theta) - \cos(\tilde{\theta})) = \frac{\delta^2}{2\,\|c - c'\|},$$

where the last equality follows from Lemma 3.2.2 and $d(\widetilde{H}, H)$ denotes the distance between the two parallel hyperplanes. See Figure 3.1 for a sketch. Since $p \in \partial B$, p belongs to $B(\sigma')^{+\delta}$ if and only if p lies in the strip bounded by H and $\widetilde{H}$, which is equivalent to

$$d(p, H) = D(p, \sigma) < \frac{\delta^2}{2\|c - c'\|}.$$

The result now follows. □

We can make this bound independent of the simplices considered, as shown in Lemma 3.2.4.

Lemma 3.2.4 *Let $\mathcal{P}$ be a δ-power protected (ε, μ)-sample. The protection parameter ι is given by $\delta = \iota\varepsilon$. For any two adjacent Voronoi vertices c and c' of $\mathrm{Vor}(\mathcal{P})$, we have*

$$\|c - c'\| \geq \frac{\delta^2}{4\varepsilon} = \frac{\iota^2\varepsilon}{4}.$$

Proof. For any simplex σ, we have $D(p, \sigma) \leq 2R_\sigma$ for all $p \in \sigma$, where R_σ denotes the radius of the circumsphere of σ. For any σ in the triangulation of an ε-net, we have $R_\sigma \leq \varepsilon$. Thus $D(p, \sigma) \leq 2\varepsilon$, and Lemma 3.2.3 yields $\|c - c'\| \geq \frac{\delta^2}{4\varepsilon}$. □

Remark 3.2.5 *In this section, we have computed a lower bound on the distance between two (adjacent) Voronoi vertices. Later on in this chapter, we shall show that Voronoi vertices are stable under metric perturbations, meaning that when a metric field is slightly modified, the position of a Voronoi vertex does not move too much. The combination of this separation and stability will then be the basis of many proofs in this thesis.*

3.2.2 Separation of Voronoi faces

Similar results can be obtained on the distance between a Voronoi vertex c and faces that are not incident to c, also referred to as *foreign* faces. Note that we are still in the context of an Euclidean metric.

The following lemma can be found in the context of ordinary protection in [21, Lemma 3.3].

Lemma 3.2.6 *Suppose that c is the circumcenter of a δ-power protected simplex σ of a Delaunay triangulation built from an ε-sample, then all foreign Voronoi faces are at least $\frac{\delta^2}{8\varepsilon}$ removed from c. We denote by p_0 an (arbitrary) vertex of σ, and by q a vertex that is not in σ but is adjacent to p_0 in* $\mathrm{Del}(\mathcal{P})$.

Proof. Let x be a point in $B(c, r) \cap \mathrm{V}_{\mathbb{E}}(p_0)$, with $0 < r < \delta^2/4\epsilon$. The upper bound for r is chosen with Lemma 3.2.3 in mind: we are trying to find a condition such that $x \in \mathrm{V}_{\mathbb{E}}(p_0)$. The notations are illustrated in Figure 3.2.

Because of the triangle inequality, we have that

$$|d(c, q) - d(x, q)| \leq d(x, c)$$
$$|d(c, p_i) - d(x, p_i)| \leq d(x, c).$$

By power protection, we have that $d(c, q)^2 \geq d(c, p_i)^2 + \delta^2$. Therefore,

$$\begin{aligned}(d(x, q) + r)^2 &\geq (d(x, p_i) - r)^2 + \delta^2 \\ d(x, q)^2 + 2rd(x, q) &\geq d(x, p_i)^2 - 2rd(x, p_i) + \delta^2 \\ d(x, q)^2 &\geq d(x, p_i)^2 - 2r(d(x, p_i) + d(x, q)) + \delta^2.\end{aligned}$$

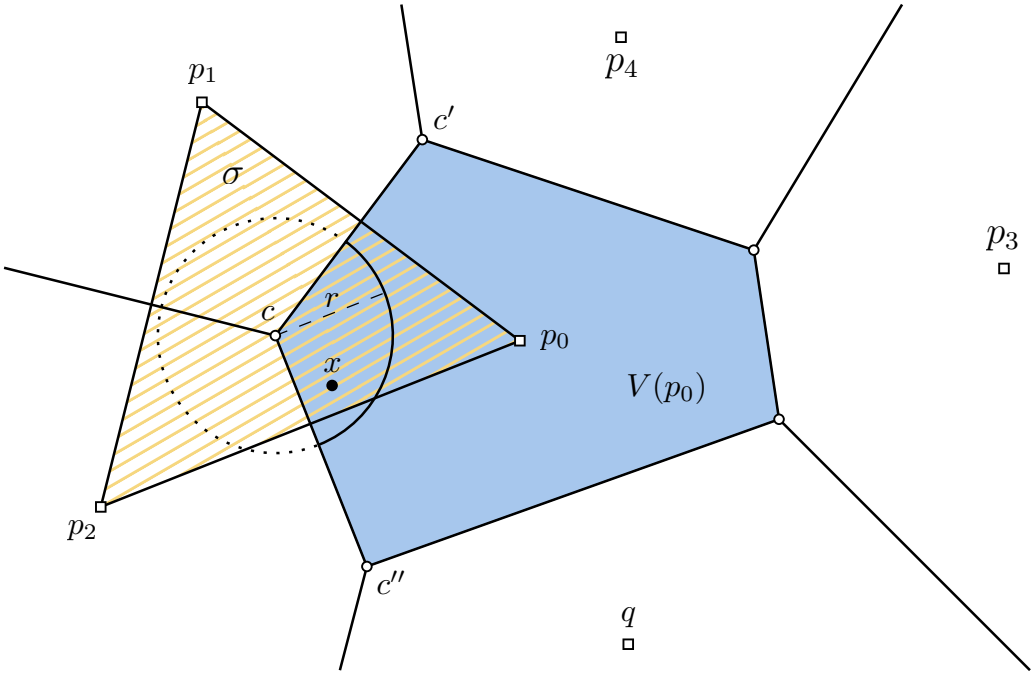

Figure 3.2: Illustration of the notations for the proof of Lemma 3.2.6. The simplex σ is dashed in beige and has vertices $p_0p_1p_2$. The distances cc' and cc'' are lower bounded by $\delta^2/4\epsilon$.

Without loss of generality, we can assume that q is the seed closest to c and thus $d(x,q) < d(x,c)$. If $\mathcal{P}$ is an ε-net, we have

$$d(x,p_i) + d(x,q) \leq d(x,p_i) + d(c,q) \leq \varepsilon + 3\varepsilon = 4\varepsilon,$$

so

$$d(x,q)^2 \geq d(x,p_i)^2 - 8r\varepsilon + \delta^2.$$

This implies that as long $r < \delta^2/8\varepsilon$, x lies in a Voronoi object associated to the vertices p_i of σ. □

Further progressing, we can show that Voronoi faces are thick, with Lemma 3.2.7. This property is useful to construct a triangulation that satisfies the hypotheses of Sperner's lemma (see Chapter III.8).

Lemma 3.2.7 *Let $\mathcal{P}$ be a δ-power protected (ε,μ)-net. Let V_0 be the Voronoi cell of the seed $p_0 \in \mathcal{P}$ in the Euclidean Voronoi diagram $\mathrm{Vor}_{\mathbb{E}}(\mathcal{P})$. Then for any k-face F_0 of V_0 with $k \in [1,\dots,n]$, there exists $x \in F_0$ such that*

$$d(x,\partial F_0) > \frac{\delta^2}{16\varepsilon},$$

where ∂F_0 denotes the boundary of the face F_0.

Proof. All the vertices of F_0 are circumcenters of $\mathrm{Vor}_{\mathbb{E}}(\mathcal{P})$. Consider the erosion of the face F_0 by a ball of radius $\frac{\delta^2}{16\varepsilon}$ and denote it F_0^-. If F_0^- is empty, we contradict the separation Lemma 3.2.6. □

3.3 Dihedral angles

The use of nets allows us to deduce bounds on the dihedral angles of faces of the Delaunay triangulation, as well as on the dihedral angles between adjacent faces of a Voronoi diagram. Those bounds are frequently used throughout this thesis, and specifically to prove stability of Voronoi vertices (see Section 3.4). Since we are interested in dihedral bounds in the Euclidean setting, the point set is first assumed to be a net with respect to the Euclidean metric field. We complicate matters slightly with Lemma 3.3.6 by assuming that the point set is a power protected net with respect to an arbitrary metric field that is not too far from the Euclidean metric field (in terms of distortion), and still manage to expose bounds with respect to the Euclidean distance.

3.3.1 Bounds on the dihedral angles of Euclidean Voronoi cells

Assuming that a point set is an (ε, μ)-net allows us to deduce lower and upper bounds on the dihedral angles between adjacent Voronoi faces when the metric field is Euclidean.

Lemma 3.3.1 *Let $\Omega = \mathbb{R}^n$ and $\mathcal{P}$ be an (ε, μ)-net with respect to the Euclidean distance on Ω. Let $p \in \mathcal{P}$ and $\mathrm{V}(p)$ be the Voronoi cell of $p \in \mathcal{P}$. Let q, $r \in \mathcal{P}$ be two seeds such that $\mathrm{V}(q)$ and $\mathrm{V}(r)$ are adjacent to $\mathrm{V}(p)$ in the Euclidean Voronoi diagram of $\mathcal{P}$. Let θ be the dihedral angle between $\mathrm{BS}(p, q)$ and $\mathrm{BS}(p, r)$. Then*

$$2 \arcsin\left(\frac{\mu}{2\varepsilon}\right) \leq \theta \leq \pi - \arcsin\left(\frac{\mu}{2\varepsilon}\right).$$

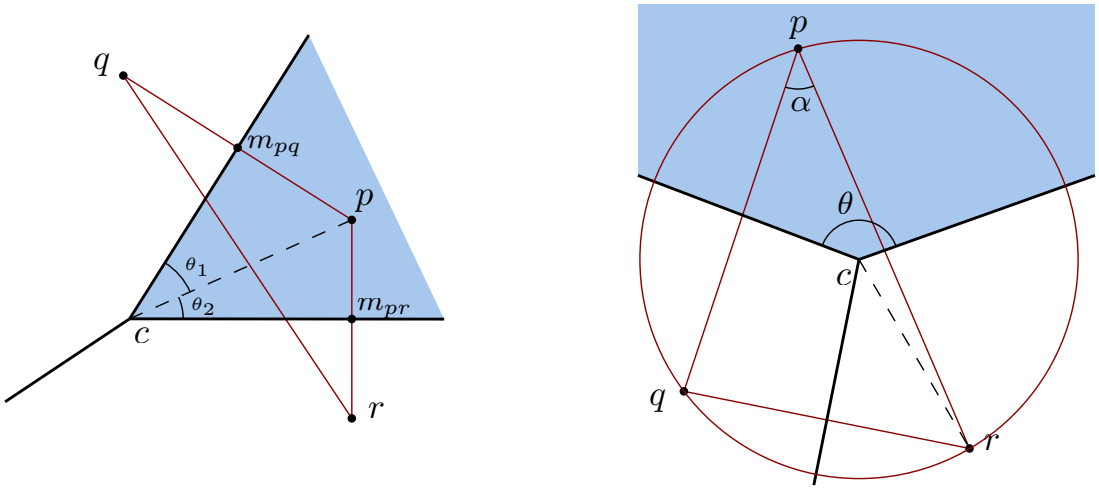

Figure 3.3: Construction and notations used in Lemma 3.3.1.

Proof. We consider the plane $\mathcal{H}$ that passes through the seeds p, q and r. Notations used below are illustrated in Figure 3.3.

Lower bound. Let m_{pq} and m_{pr} be the projections of the seed p on respectively the bisectors $\mathrm{BS}(p, q)$ and $\mathrm{BS}(p, r)$. Since $\mathcal{P}$ is an (ε, μ)-net, we have that $l_q = |p - m_{pq}| \geq \mu/2$, $l_r = |p - m_{pr}| \geq \mu/2$ and $L = |p - c| \leq \varepsilon$. Thus

$$\theta = \arcsin\left(\frac{l_q}{L}\right) + \arcsin\left(\frac{l_r}{L}\right) \geq 2 \arcsin\left(\frac{\frac{\mu}{2}}{\varepsilon}\right) = 2 \arcsin\left(\frac{\mu}{2\varepsilon}\right).$$

Note that since $0 < \mu < 2\varepsilon$, we have $0 < \mu/2\varepsilon < 1$.

Upper bound. To obtain an upper bound on θ, we compute a lower bound on the angle $\alpha = \widehat{qpr}$ at p, noting that $\theta = \pi - \alpha$. Let $l_{qr} = |q - r|$, and $R = |c - r|$. By the law of sines, we have

$$\frac{l_{qr}}{\sin(\alpha)} = 2R.$$

Since $\mathcal{P}$ is an (ε, μ)-net, we have $l_{qr} \geq \mu$ and $R \leq \varepsilon$. Finally,

$$\alpha \geq \arcsin\left(\frac{\mu}{2\varepsilon}\right) \Longrightarrow \theta \leq \pi - \arcsin\left(\frac{\mu}{2\varepsilon}\right).$$

□

3.3.2 Bounds on the dihedral angles of Euclidean Delaunay simplices

Bounds on the dihedral angles of simplices guarantee the thickness – the smallest height of any vertex – of simplices, and thus their quality. Additionally, they can be used to show that circumcenters of adjacent simplices are far from one another, thus proving the stability of circumcenters and of Delaunay simplices.

Using power protection with respect to the Euclidean metric field

We first assume that the metric field g is the Euclidean metric field $g_{\mathbb{E}}$ and show that the simplices of an Euclidean Delaunay triangulation constructed from a power-protected net are thick.

Recall that the dihedral angle can be expressed through heights as

$$\sin \angle(\mathrm{Aff}(\sigma_p), \mathrm{Aff}(\sigma_q)) = \frac{D(p,\sigma)}{D(p,\sigma_q)} = \frac{D(q,\sigma)}{D(q,\sigma_p)}.$$

The bound on dihedral angles is obtained by bounding the height of vertices in a simplex. An obvious upper bound on the height of a vertex p in σ is $D(p,\sigma) < 2\varepsilon$. A lower bound was already obtained in Lemma 3.2.3: we have that $D(p,\sigma) \geq \delta^2/2 \left\|c - c'\right\| = \delta^2/4\varepsilon$. The following lemma is more technical but provides a tighter bound. The proof is similar to that of Lemma [19, Lemma 3.3], but our notions of metric distortion differ and power protection is here assumed instead of protection.

Lemma 3.3.2 *Let $\mathcal{P}$ be an (ε, μ)-net with respect to the Euclidean metric field $g_{\mathbb{E}}$. Let $\sigma = q * \tau$ be a simplex of* $\mathrm{Del}(\mathcal{P})$*. Suppose also that $\tau \leq \sigma'$ and that σ is not a face of σ'. Denote $B = B_{\mathbb{R}^n}(c, r)$ and $B' = B_{\mathbb{R}^n}(c', r')$ the respective Delaunay balls of σ and σ', and $\mathcal{H} = \mathrm{Aff}(\partial B \cap \partial B')$. The protection parameter ι is given by $\delta = \iota\varepsilon$ and the separation parameter λ is given by $\mu = \lambda\varepsilon$. If σ' is δ-power protected with respect to $g_{\mathbb{E}}$, then*

$$d(q, \mathcal{H}) > A_{\lambda,\iota}\varepsilon,$$

where $A_{\lambda,\iota}$ is a constant that depends on the separation and protection parameters λ and ι and will be detailed in the proof. It follows that if $\mathcal{P}$ is δ-power protected, then

$$D(q, \sigma) > A_{\lambda,\iota}\varepsilon.$$

Proof. We first look to bound the distance $d(q, \mathcal{H})$. Everything will be performed in the plane $\mathcal{Q}$ spanned by q, c and c' (see Figure 3.4). This plane is orthogonal to the $(n-1)$-flat $\mathcal{H}$ and thus the projection q^* of q onto $\mathcal{H}$ is such that $d(q, q^*) = d(q, \mathcal{H})$.

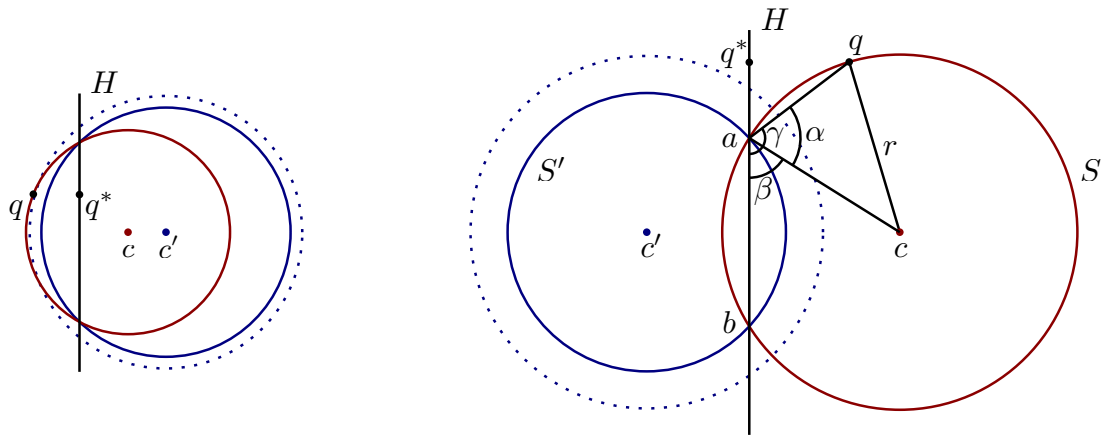

Figure 3.4: Construction and notations for the Lemma 3.3.2.

If $\mathcal{H}$ separates q from c (as in Figure 3.4, left), then $\partial B'$ separates q from q^* and $d(q,q^*) > d(q,\partial B')$. Let $q_{B'}$ be the point on $\partial B'$ that realizes the distance $d(q,\partial B')$. Since σ' is δ-power protected, we have

$$\begin{aligned} d(q,q_{B'}) = d(c',q) - d(c',q_{B'}) &\geq \sqrt{r'^2+\delta^2} - r' \\ &\geq r'\sqrt{1+\frac{\delta^2}{r'^2}} - r'. \end{aligned}$$

To simplify computations, we wish to find a bound on $d(q,q_{B'})$ that is proportional to r'. Therefore, we seek to express $\sqrt{1+\frac{\delta^2}{r'^2}}$ as $r + f(\delta, r')$. By concavity of $f(x) = \sqrt{1+x}$, we have on $[0;x_0]$ that

$$(1+x)^{\frac{1}{2}} \geq 1 + \frac{(1+x_0)^{\frac{1}{2}} - 1}{x_0} x. \tag{3.6}$$

Figure 3.5 illustrates the inequality with a graph of the two functions involved in Inequality 3.6.

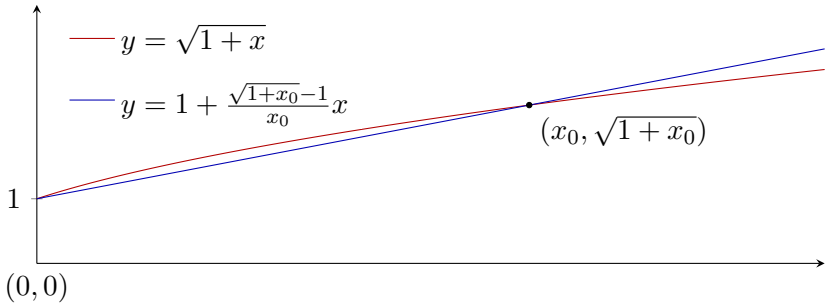

Figure 3.5: Illustration of Inequality 3.6.

Since $\delta = \iota\varepsilon$ and $\mu/2 \leq r' \leq \varepsilon$, we have

$$\frac{\delta^2}{r'^2} \leq \frac{4\iota^2\varepsilon^2}{\mu^2} = \frac{4\iota^2}{\lambda^2}.$$

In the following, we use $\alpha = \frac{4\iota^2}{\lambda^2}$.

Thus,

$$\left(1+\frac{\delta^2}{r'^2}\right)^{\frac{1}{2}} \geq 1+\frac{\sqrt{1+\alpha}-1}{\alpha}\frac{\delta^2}{r'^2}.$$

Let $\nu := \frac{1}{\alpha}\left(\sqrt{1+\alpha}-1\right)$. We have

$$d(q,q_{B'}) \geq r'\sqrt{1+\frac{\delta^2}{r'^2}} - r' \geq r'\left(1+\nu\frac{\delta^2}{r'^2}\right) - r' = \nu\frac{\delta^2}{r'}.$$

Finally, since $r' \leq \varepsilon$ we have $d(q,\mathcal{H}) \geq \nu\frac{\delta^2}{\varepsilon} = \nu\iota^2\varepsilon$.

Assume now that q and c lie on the same side of $\mathcal{H}$ (as in Figure 3.4, right). Let $S' = \mathcal{Q}\cap\partial B'$ and $S = \mathcal{Q}\cap\partial B$. Let a and b be the points of intersection $S\cap S'$.

We bound $d(q,\mathcal{H})$ by exhibiting a bound on the angle $\gamma = \widehat{qab}$. Without loss of generality, we assume that $\gamma \geq \widehat{qba}$. We can assume that $\gamma \geq \frac{\pi}{2}$, otherwise $q^* = q_{B'}$ and we have immediately $d(q,\mathcal{H}) = d(q,\partial B') \geq \nu\frac{\delta^2}{\varepsilon}$.

Since $d(a,q) \geq \nu\frac{\delta^2}{\varepsilon}$, we have $d(q,q^*) = d(a,q)\sin(\pi-\gamma) \geq \nu\frac{\delta^2}{\varepsilon}\sin(\gamma)$. Let $\alpha = \widehat{qac}$ and $\beta = \widehat{cab}$. We have

$$\cos(\alpha) = \frac{d(a,q)}{2r} \geq \frac{\nu}{2}\frac{\delta^2}{\varepsilon^2} = \frac{\nu\iota^2}{2},$$

thus $\alpha \leq \arccos\left(\frac{\nu}{2}\frac{\delta^2}{\varepsilon^2}\right) \leq \frac{\pi}{2}$. On the other hand, we have $|cc'| \geq \frac{\delta^2}{4\varepsilon}$ by Lemma 3.2.4. Similarly, we deduce that $\beta \leq \arcsin\left(\frac{\iota^2}{8}\right) \leq \frac{\pi}{2}$. Combining both results, we have $\frac{\pi}{2} \leq \gamma = \alpha+\beta \leq \gamma'$ with

$$\gamma' = \arccos\left(\frac{\nu\iota^2}{2}\right) + \arcsin\left(\frac{\iota^2}{8}\right).$$

Since $\frac{\pi}{2} \geq \gamma \geq \gamma' \geq \pi$,

$$\begin{aligned}
d(q,\mathcal{H}) &\geq \nu\frac{\delta^2}{\varepsilon}\sin(\gamma) \geq \nu\frac{\delta^2}{\varepsilon}\sin(\gamma') \\
&= \nu\frac{\delta^2}{\varepsilon}\sin\left(\arccos\left(\frac{\nu\iota^2}{2}\right) + \arcsin\left(\frac{\iota^2}{8}\right)\right) \\
&\geq \nu\frac{\delta^2}{\varepsilon}\left[\sin\left(\arccos\left(\frac{\nu\iota^2}{2}\right)\right)\cos\left(\arcsin\left(\frac{\iota^2}{8}\right)\right) + \frac{\nu\iota^2}{2}\frac{\iota^2}{8}\right] \\
&\geq \nu\frac{\delta^2}{\varepsilon}\left[\sqrt{1-\left(\frac{\nu\iota^2}{2}\right)^2}\sqrt{1-\left(\frac{\iota^2}{8}\right)^2} + \frac{\nu\iota^4}{16}\right] \\
&\geq A_{\lambda,\iota}\,\varepsilon,
\end{aligned}$$

with

$$A_{\lambda,\iota} = \left[\sqrt{1-\left(\frac{\nu\iota^2}{2}\right)^2}\sqrt{1-\left(\frac{\iota^2}{8}\right)^2} + \frac{\nu\iota^4}{16}\right]\nu\iota^2.$$

Since $\mathrm{Aff}(\tau) \subset \mathcal{H}$, it follows that $D(q,\sigma) \geq d(q,\mathcal{H})$. □

This new bound is not always tighter. In fact, $A_{\lambda,\iota}$ is only greater than $\iota^2/4$ when $\sqrt{\iota^2/2} \leq \lambda$ (value obtained by direct computation). In a generic configuration, this condition tends to be satisfied and we therefore shall use the bound using $A_{\lambda,\iota}$ in the rest of this thesis. Note that the choice of the bound has no consequence on the results further deduced. The comparison between both bounds $\iota^2/4$ and $A_{\lambda,\iota}$ is shown in Figure 3.6.

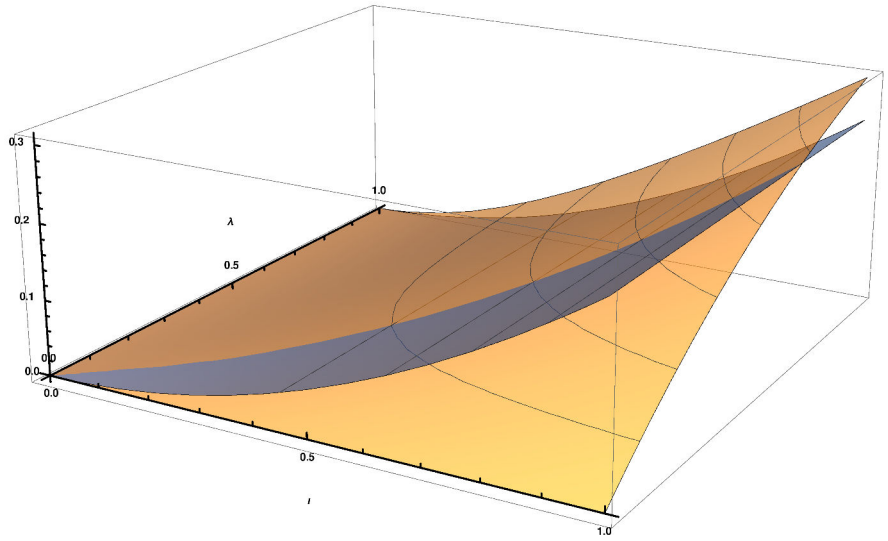

Figure 3.6: Comparison between the bounds provided by Lemmas 3.2.3 and 3.3.2. $\iota^2/4$ is drawn in blue, and $A_{\lambda,\iota}$ in orange.

Lemma 3.3.3 *Let $\mathcal{P}$ be a δ-power protected (ε, μ)-net with respect to the Euclidean metric field g_0. Let φ be the dihedral angle between two facets τ_1 and τ_2 of a simplex σ of $\mathrm{Del}_{g_0}(\mathcal{P})$. Then*

$$\arcsin(s_0) \leq \varphi \leq \pi - \arcsin(s_0),$$

with $s_0 = \frac{A_{\lambda,\iota}}{2}$ and $A_{\lambda,\iota}$ defined as in the previous Lemma.

Proof. This is a consequence from Lemma 3.3.2, which gives

$$D(q, \sigma_p) \geq D(q, \sigma) \geq A_{\lambda,\iota}\, \varepsilon,$$

and from the observation that $D(q, \sigma) \leq 2\varepsilon$. We have $\varphi = \angle(\mathrm{Aff}(\sigma_p), \mathrm{Aff}(\sigma_q))$. Recall that

$$\sin(\varphi) = \sin \angle(\mathrm{Aff}(\sigma_p), \mathrm{Aff}(\sigma_q)) = \frac{D(p, \sigma)}{D(p, \sigma_q)} = \frac{D(q, \sigma)}{D(q, \sigma_p)}$$

Therefore

$$\sin \angle(\mathrm{Aff}(\sigma_p), \mathrm{Aff}(\sigma_q)) \geq \frac{A_{\lambda,\iota}\, \varepsilon}{2\varepsilon} = \frac{A_{\lambda,\iota}}{2} =: s_0$$

Note that $0 < s_0 < 1$ and thus

$$\arcsin(s_0) \leq \varphi \leq \pi - \arcsin(s_0).$$

□

Using power protection with respect to an arbitrary metric field

When considering a Voronoi diagram built using the geodesic distance induced by an arbitrary metric field g, the assumption of a power protected net is naturally made with respect to this geodesic distance. To prove the stability of the power protected assumption under metric perturbation, we will however need to deduce lower and upper bounds on the dihedral angles between faces of the simplices of the Delaunay complex with respect to the Euclidean metric field . We prove here that if the point set $\mathcal{P}$ is a δ-power protected (ε, μ)-net with respect to an arbitrary metric field g and if the distortion between g and the Euclidean metric field $g_{\mathbb{E}}$ is small, then the dihedral angles of the simplices of the Euclidean Delaunay triangulation of $\mathcal{P}$ can be bounded.

We first give a result on the stability of Delaunay balls which expresses that if two metric fields have low distortion, the Delaunay balls of a simplex with respect to each metric field are close. Naturally, one of those metric fields will then be taken as the Euclidean metric field. A similar result can be found in the proof of Lemma 5 in the theoretical analysis of locally uniform anisotropic meshes of Boissonnat et al. [31].

Lemma 3.3.4 *Let $U \subset \Omega$ be open, and g and g' be two Riemannian metric fields on U. Let $\psi_0 \geq 1$ be a bound on the metric distortion. Suppose that U is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$, such that $\forall p \in B(p_0, r_0), \psi(g(p), g'(p)) \leq \psi_0$. Assume furthermore that g' is the Euclidean distance (thus $d_{g'} = d_{\mathbb{E}}$). Let $B = B_g(c, r)$ be the geodesic ball with respect to the metric field g, centered on $c \in U$ and of radius r. Assume that $B_{\mathbb{E}}(c, \psi_0 r) \subset U$. Then B can be encompassed by two Euclidean balls $B_{\mathbb{E}}(c, r_{-\psi_0})$ and $B_{\mathbb{E}}(c, r_{+\psi_0})$ with $r_{-\psi_0} = r/\psi_0$ and $r_{+\psi_0} = \psi_0 r$.*

Proof. This is a straight consequence from Lemma 3.1.4. Indeed, we have for all $x \in U$ that

$$\frac{1}{\psi_0} d_{\mathbb{E}}(c, x) \leq d_g(c, x) \leq \psi_0 d_{\mathbb{E}}(c, x),$$

and similarly

$$\frac{1}{\psi_0} d_g(c, x) \leq d_{\mathbb{E}}(c, x) \leq \psi_0 d_g(c, x).$$

Thus,

$$x \in B_{\mathbb{E}}\left(c, \frac{r}{\psi_0}\right) \iff d_{\mathbb{E}}(c, x) \leq \frac{r}{\psi_0} \Longrightarrow \frac{1}{\psi_0} d_g(c, x) \leq \frac{r}{\psi_0},$$

giving us $B_{\mathbb{E}}(c, r_{-\psi_0}) \subset B$. On the other hand, we have

$$x \in B_g(c, r) \iff d_g(c, x) \leq r \Longrightarrow \frac{1}{\psi_0} d_{\mathbb{E}}(c, x) \leq r,$$

giving us $B \subset B_{\mathbb{E}}(c, r_{+\psi_0})$. □

Note that $r_{-\psi_0}$ and $r_{+\psi_0}$ go to r as ψ_0 goes to 1.

We now use this stability result to provide Euclidean dihedral angle bounds assuming power protection with an arbitrary metric field that is close to $g_{\mathbb{E}}$. We first require the intermediary result given by Lemma 3.3.5.

Lemma 3.3.5 *Let H be a hyperplane in Euclidean n-space and τ an $n-1$-simplex whose vertices lie at most η from the H and whose minimum height is h_{min}. Then the angle ξ between $\mathrm{Aff}(\tau)$ and H is bounded from above by*

$$\sin(\xi) \leq \frac{\eta d}{h_{min}}.$$

Proof. By definition, the barycenter of a $(n-1)$-simplex has barycentric coordinates $\lambda_i = 1/d$. This means that it has distance a $h_{\min}/n$ to each of its faces. So the ball centered on the barycenter with radius $h_{\min}/n$ is contained in the simplex. This means that for any direction in $\mathrm{Aff}(\tau)$ there exists a line segment of length $2h_{\min}/n$ that lies within τ. Moreover the end points of this line segments lie at most η from H because the vertices of the τ do. This means that the angle ξ is bounded by

$$\sin(\xi) \leq \frac{2\eta}{2h_{\min}/d} = \frac{\eta d}{h_{\min}}.$$

□

We can now give the main result which bounds *Euclidean* dihedral angles, assuming power protection with respect to the arbitrary metric field.

Lemma 3.3.6 *Let $U \subset \Omega$ be open, and g and g' be two Riemannian metric fields on U. Let $\psi_0 \geq 1$ be a bound on the metric distortion. Suppose that U is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$, such that $\forall p \in B(p_0, r_0), \psi(g(p), g'(p)) \leq \psi_0$. Let $\mathcal{P}_U$ be a δ-power protected (ε, μ)-net over U, with respect to g. Let $B = B_g(c, r)$ and $B' = B_g(c', r')$ be the geodesic Delaunay balls of $\sigma = \tau * p$ and $\sigma' = \tau * p'$, with $\sigma, \sigma' \in Del_g(\mathcal{P}_U)$. Assume that $\mathcal{P}_U$ is sufficiently dense such that U contains B and B'. Then the minimum height of the simplex satisfies*

$$h_{min} = \sqrt{1 - \left(n \frac{(r^2 + r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right)}{8\varepsilon h_{min}} \right)^2} \cdot \left(\frac{(r^2 + r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right) + \frac{\delta^2}{\psi_0^2}}{4\varepsilon} - \frac{n \frac{(r^2+r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right)}{8\varepsilon h_{min}}}{\sqrt{1 - \left(n \frac{(r^2+r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right)}{8\varepsilon h_{min}} \right)^2}} (r + r') \right).$$

Note that this is the height of p in σ with respect to the Euclidean metric.

We proceed in a similar fashion as the proof Lemma 3.2.1. However, a significant difference is that we are interested here in only proving that power protection with respect to the generic metric field provides a height bound in the Euclidean setting (rather than an equivalence).

Proof. We use the following notations, illustrated in Figure 3.7:

- $B_{\mathbb{E}}^{\pm\psi_0}$ and $B_{\mathbb{E}}^{\prime\pm\psi_0}$ are the two sets of (Euclidean) enclosing balls of respectively B and B' defined as in Lemma 3.3.4.
- $B'^{+\delta}$ is the power protected ball of σ', given by $B'^{+\delta} = B(c', \sqrt{r'^2 + \delta^2})$.
- $B_{\mathbb{E}}^{\prime+\delta,\pm\psi_0}$ are the two Euclidean balls enclosing $B'^{+\delta}$.
- $S = \partial B^{-\psi_0} \cap \partial B'^{+\psi_0}$, $\tilde{S} = \partial B^{+\psi_0} \cap \partial B'^{+\delta,-\psi_0}$ and $S' = \partial B^{+\psi_0} \cap \partial B'^{-\psi_0}$.
- q is a point on S, $\tilde{q}$ is a point on $\widetilde{S}$ and q' is a point on S'.
- H is the geodesic supporting plane of τ, that is $\{\arg\min(\sum_{v_i \in \tau} \lambda_i d_g(x, v_i))\}$.

- $H_{\mathbb{E}}$, $\widetilde{H}_{\mathbb{E}}$ and $H_{\mathbb{E}}$ are the two Euclidean hyperplanes orthogonal to $[cc']$ passing through respectively q, $\tilde{q}$ and q'.

- $\theta = \widehat{c'cq}$ $\widetilde{\theta} = \widehat{c'c\tilde{q}}$, and $\lambda_c = |cc'|/r$.

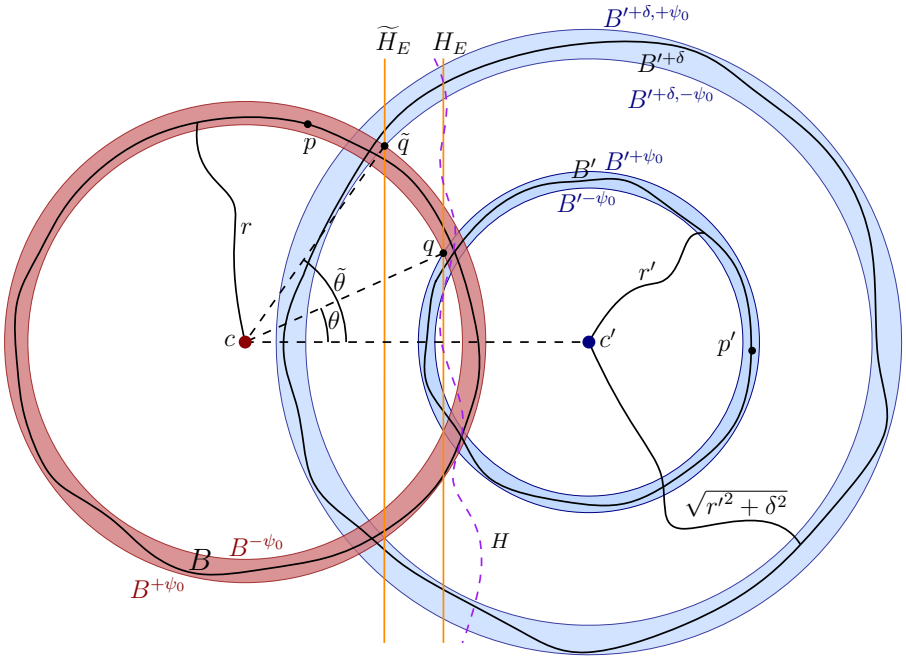

Figure 3.7: Construction and notations used in the proof of Lemma 3.3.6

While the vertices of τ live on H, $\mathrm{Aff}_{\mathbb{E}}(\tau)$ is not necessarily orthogonal to $[cc']$ and consequently

$$d_{\mathbb{E}}(p, H_{\mathbb{E}}) \leq d_{\mathbb{E}}(p, \tau).$$

The separation between the hyperplanes $H_{\mathbb{E}}$ and $\widetilde{H}_{\mathbb{E}}$ provides a lower bound on the distance $d_{\mathbb{E}}(p, \tau)$, thus on the height $D_{\mathbb{E}}(p, \sigma)$. We therefore seek to bound $d_{\mathbb{E}}(H_{\mathbb{E}}, \widetilde{H}_{\mathbb{E}})$.

By definition of the enclosing Euclidean balls, we have that

$$|cq| = \frac{r}{\psi_0}, \quad |c\tilde{q}| = \psi_0 r, \quad |c'q| = \psi_0 r', \text{ and } |c'\tilde{q}| = \frac{1}{\psi_0}\sqrt{r'^2+\delta^2}$$

Using the law of cosines in the triangles $\triangle cc'q$ and $\triangle cc'\tilde{q}$, we find

$$\frac{r'^2+\delta^2}{\psi_0^2} = \lambda_c^2 r^2 + \psi_0^2 r^2 - 2\lambda_c \psi_0 r^2 \cos(\widetilde{\theta})$$
$$\psi_0^2 r'^2 = \lambda_c^2 r^2 + \frac{r^2}{\psi_0^2} - 2\frac{\lambda_c r^2}{\psi_0}\cos(\theta),$$

where $\lambda_c r = |c - c'|$. Subtracting one from the other, we obtain

$$\begin{aligned}
\psi_0^2 r'^2 - \frac{r'^2 + \delta^2}{\psi_0^2} &= \frac{r^2}{\psi_0^2} - \psi_0^2 r^2 + 2\lambda_c \psi_0 r^2 \cos(\widetilde{\theta}) - 2\frac{\lambda_c r^2}{\psi_0}\cos(\theta) \\
\iff \frac{r}{\psi_0}\cos(\theta) - \psi_0 r\cos(\widetilde{\theta}) &= \frac{\frac{r^2}{\psi_0^2} - \psi_0^2 r'^2 + \frac{r'^2+\delta^2}{\psi_0^2} - \psi_0^2 r^2}{2\lambda_c r} \\
\iff \frac{r}{\psi_0}\cos(\theta) - \psi_0 r\cos(\widetilde{\theta}) &= \frac{(r^2 + r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right) + \frac{\delta^2}{\psi_0^2}}{2\lambda_c r},
\end{aligned}$$

so that

$$d(H_{\mathbb{E}}, \widetilde{H}_{\mathbb{E}}) = \frac{r}{\psi_0}\cos(\theta) - \psi_0 r\cos(\widetilde{\theta}) = \frac{(r^2 + r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right) + \frac{\delta^2}{\psi_0^2}}{2\lambda_c r}.$$

Similarly we can calculate the distance $d_{\mathbb{E}}(H_{\mathbb{E}}, H'_{\mathbb{E}})$ to be:

$$\begin{aligned}
d_{\mathbb{E}}(H_{\mathbb{E}}, H'_{\mathbb{E}}) &= \frac{\frac{1}{\psi^2}(r')^2 + |c - c'|^2 - \psi^2 r^2}{2|c - c'|} - \frac{\psi^2 (r')^2 + |c - c'|^2 - \frac{1}{\psi^2} r^2}{2|c - c'|} \\
&= \frac{(r^2 + r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right)}{2|c - c'|}. \\
&\leq \frac{(r^2 + r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right)}{4\varepsilon}
\end{aligned}$$

Lemma 3.3.5 gives us that the angle ξ between $H_{\mathbb{E}}$ and $\mathrm{Aff}(\tau)$ is bounded by

$$\sin(\xi) \leq \frac{n\, d_{\mathbb{E}}(H_{\mathbb{E}}, H'_{\mathbb{E}})}{2h_{\min}}.$$

The vertices of τ lie in between $H_{\mathbb{E}}$ and $H'_{\mathbb{E}}$ and inside $B_{\mathbb{E}}(c', \psi r')$. Thus, if we restrict to the $\tilde{q}cc'$ plane, distance between the point where $\mathrm{Aff}(\tau)$ intersects $H_{\mathbb{E}}$ and the orthogonal projection $\pi_{H_{\mathbb{E}}}(\tilde{q})$ of $\tilde{q}$ on $H_{\mathbb{E}}$ is at most $\psi(r + r')$. This in turn implies that the line connecting $\pi_{H_{\mathbb{E}}}(\tilde{q})$ and $\tilde{q}$ intersects $\mathrm{Aff}(\tau)$ at most distance $(r + r')\tan(\xi)$ from $H_{\mathbb{E}}$. This also gives us that the distance from $\tilde{q}$ to its orthogonal projection $\pi_{\mathrm{Aff}(\tau)}(\tilde{q})$ on $\mathrm{Aff}(\tau)$ is bounded by

$$\begin{aligned}
\cos(\xi)(d(\tilde{H}_{\mathbb{E}}, H_{\mathbb{E}}) - \tan(\xi)(r + r')) =& \sqrt{1 - \left(\frac{n\, d_{\mathbb{E}}(H_{\mathbb{E}}, H'_{\mathbb{E}})}{2h_{\min}}\right)^2} \cdot \\
& \left(d(\tilde{H}_{\mathbb{E}}, H_{\mathbb{E}}) - \frac{\frac{n\, d_{\mathbb{E}}(H_{\mathbb{E}}, H'_{\mathbb{E}})}{2h_{\min}}}{\sqrt{1 - \left(\frac{n\, d_{\mathbb{E}}(H_{\mathbb{E}}, H'_{\mathbb{E}})}{2h_{\min}}\right)^2}}(r + r') \right).
\end{aligned}$$

Here we note that $h_{\min}$ originally referred to the minimum height of a face, but because the height of a face is always greater than the height in the simplex we may read this in a general way, that is we regard $h_{\min}$ as a universal lower bound on the height. Because $|\tilde{q} - \pi_{\mathrm{Aff}(\tau)}(\tilde{q})|$ bounds the height of

the simplex we get the following relation:

$$h_{\min} = \sqrt{1 - \left(\frac{n\, d_{\mathbb{E}}(H_{\mathbb{E}}, H'_{\mathbb{E}})}{2h_{\min}}\right)^2} \left(d(\tilde{H}_{\mathbb{E}}, H_{\mathbb{E}}) - \frac{\frac{n\, d_{\mathbb{E}}(H_{\mathbb{E}}, H'_{\mathbb{E}})}{2h_{\min}}}{\sqrt{1 - \left(\frac{n\, d_{\mathbb{E}}(H_{\mathbb{E}}, H'_{\mathbb{E}})}{2h_{\min}}\right)^2}} (r + r') \right)$$

$$h_{\min} = \sqrt{1 - \left(n \frac{(r^2 + r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right)}{8\varepsilon h_{\min}} \right)^2} \cdot$$

$$\left(\frac{(r^2 + r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right) + \frac{\delta^2}{\psi_0^2}}{4\varepsilon} - \frac{n \frac{(r^2+r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right)}{8\varepsilon h_{\min}}}{\sqrt{1 - \left(n \frac{(r^2+r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right)}{8\varepsilon h_{\min}} \right)^2}} (r + r') \right)$$

To make the expression a bit more readable, we introduce

$$s_1 = \frac{(r^2 + r'^2)\left(\frac{1}{\psi_0^2} - \psi_0^2\right)}{4\varepsilon}$$

so that

$$h_{\min} = \sqrt{1 - \left(n \frac{s_1}{2h_{\min}} \right)^2} \left(\left(s_1 + \frac{\delta^2}{4\varepsilon\psi_0^2} \right) - \frac{n \frac{s_1}{2h_{\min}}}{\sqrt{1 - \left(n \frac{s_1}{2h_{\min}} \right)^2}} (r + r') \right)$$

$$h_{\min} + n \frac{s_1}{2h_{\min}} (r + r') = \sqrt{1 - \left(n \frac{s_1}{2h_{\min}} \right)^2} \left(s_1 + \frac{\delta^2}{4\varepsilon\psi_0^2} \right).$$

Multiplying with $h_{\min}$ and squaring we find:

$$(h_{\min}^2 + n \frac{s_1}{2}(r + r'))^2 = \left(h_{\min}^2 - \left(n \frac{s_1}{2} \right)^2 \right) \left(s_1 + \frac{\delta^2}{4\varepsilon\psi_0^2} \right)^2$$

We note that in the limit $\psi \to 1$, s_1 tends to zero, we therefore expand $h_{\min}$ is terms of s_1. Using a computer algebra system we find

$$h_{\min}^2 = \frac{\delta^4}{2^{10}\psi^4\varepsilon^2} + \left(\frac{\delta^2}{128\psi^2\varepsilon} - n(r + r') \right) s_1 - \frac{\delta^4 \left(16n^2 - 1\right) + 2^{14} n^2 \psi^4 \varepsilon^2 (r + r')^2}{64\delta^4} s_1^2 + \mathcal{O}(s_1^3)$$

We emphasize that this equation gives $h_{\min} = \delta^2/4\varepsilon$ as ψ tends to 1. This means that for sufficiently small metric distortion the height of a protected simplex will be strictly positive. □

Lemma 3.3.7 *Let g be a metric field and $\mathcal{P}$ be a point set defined over $\mathbb{R}^n$. Let $\psi_0 = \psi(g, \mathbb{E})$. Assume that $\mathcal{P}$ is a δ-power protected (ε, μ)-net over $\mathbb{R}^n$. Let ϕ be the dihedral angle between two faces of a simplex $\tau \in \mathrm{Del}_{\mathbb{E}}(\mathcal{P})$. Then*

$$\arcsin(s_0) \leq \phi \leq \pi - \arcsin(s_0),$$

with s_0 detailed in the proof.

Proof. Denote $h_{\min}$ the lower bound on $D(q, \sigma)$ obtained in the previous lemma. We also immediate have that

$$D(q, \sigma) \leq 2\varepsilon.$$

Let φ be the dihedral angle between $\mathrm{Aff}(\sigma_p)$ and $\mathrm{Aff}(\sigma_q)$. Recall that

$$\sin(\varphi) = \sin \angle(\mathrm{Aff}(\sigma_p), \mathrm{Aff}(\sigma_q)) = \frac{D(p, \sigma)}{D(p, \sigma_q)} = \frac{D(q, \sigma)}{D(q, \sigma_p)}$$

Thus

$$\sin(\varphi) \geq \frac{h_{\min}}{2\varepsilon} =: s_0.$$

For s_0 to make sense, we want $s_0 > 0$, which is bound to happen as ψ_0 goes to 1, as shown in Lemma 3.3.6: $h_{\min}$ goes to $\delta^2/4\varepsilon$ thus $h_{\min}/4\varepsilon$ goes to $\iota^2/4 > 0$. □

3.4 Stability

The notion of stability designates the conservation of a property despite changes of other parameters. In our context, the main assumptions concern the nature of point sets: we assume that point sets are power protected nets and wish to preserve these hypotheses despite (small) metric perturbations. The stability is important both from a theoretical and a practical point of view. Indeed, if an assumption is stable under perturbation, we can simplify matters without losing information. For example, we will prove that if a point set is a net with respect to a metric field g, then it is also a net (albeit with slightly different sampling and separation parameters) for a metric field g' that is close to g (in terms of distortion) In a practical context, the stability of an assumption provides robustness with respect to numerical errors (see, for example, the work of Funke et al. [74] on the stability of Delaunay simplices).

3.4.1 Stability of the protected net hypothesis under metric transformation

The square root of a metric allows us to switch between the embedding space of the domain endowed with the Euclidean metric and the metric space endowed with the Euclidean metric. It is rather immediate that the power protected net property is preserved when the point set is transformed between these spaces, as shown by the following lemma.

Lemma 3.4.1 *Let $\mathcal{P}$ be a δ-power protected (ε, μ)-net in Ω. Let $g = g_0$ be a uniform Riemannian metric field and F_0 a square root of g_0. If $\mathcal{P}$ is a δ-power protected (ε, μ)-net with respect to g_0 then $\mathcal{P}' = \{F_0 p_i, p_i \in \mathcal{P}\}$ is a δ-power protected (ε, μ)-net with respect to the Euclidean metric.*

Proof. This results directly from the observation that

$$d_0(x, y)^2 = (x - y)^t g_0 (x - y) = \|F_0(x - y)\|^2 = d(F_0 x, F_0 y)^2.$$

□

3.4.2 Stability of the protected net hypothesis under metric perturbation

Metric field perturbations are small modifications of a metric field in terms of distortion. Since a generic Riemannian metric field g is difficult to study, we will generally consider a uniform approximation g_0 of g within a small neighborhood, such that the distortion between both metric fields is small over that neighborhood. In that context, the perturbation of g is the act of bringing g onto g_0. Stability of the assumption of power protection was previously investigated by Boissonnat et al. [22] in the context of manifold reconstructions.

Stability of the net property

The following lemma shows that the "net" property is preserved when the metric field is perturbed: a point set that is a net with g is also a net with respect to g', a metric field that is close to g.

Lemma 3.4.2 *Let $U \subset \Omega$ be open, and g and g' be two Riemannian metric fields on U. Let $\psi_0 \geq 1$ be a bound on the metric distortion. Suppose that U is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$, such that $\forall p \in B(p_0, r_0), \psi(g(p), g'(p)) \leq \psi_0$. Let $\mathcal{P}_U$ be a finite point set in U. Suppose that $\mathcal{P}_U$ is an (ε, μ)-net of U with respect to g. Then $\mathcal{P}_U$ is an $(\varepsilon_{g'}, \mu_{g'})$-net of U with respect to g' with $\varepsilon_{g'} = \psi_0\varepsilon$ and $\mu_{g'} = \mu/\psi_0$.*

Proof. By Lemma 3.1.4, we have that

$$\forall x, y \in U, \frac{1}{\psi_0} d_{g'}(x, y) \leq d_g(x, y) \leq \psi_0 d_{g'}(x, y).$$

Therefore

$$\forall x \in U, \exists p \in \mathcal{P}_U, d_g(x, p) \leq \varepsilon \Rightarrow \forall x \in U, \exists p \in \mathcal{P}_U, d_{g'}(x, p) \leq \psi_0\varepsilon,$$

and

$$\forall p, q \in \mathcal{P}_U, d_g(p, q) \geq \mu \Rightarrow \forall p, q \in \mathcal{P}_U, d_{g'}(p, q) \geq \frac{\mu}{\psi_0}.$$

And, with $\varepsilon_{g'} = \psi_0\varepsilon$ and $\mu_{g'} = \mu/\psi_0$, $\mathcal{P}$ is an $(\varepsilon_{g'}, \mu_{g'})$-net of U. □

Remark 3.4.3 *We assumed that $\mu = \lambda\varepsilon$. By Lemma 3.4.2, we have $\varepsilon' = \psi_0\varepsilon$ and $\mu' = \frac{\mu}{\psi_0}$. Therefore*

$$\mu' = \frac{\lambda\varepsilon}{\psi_0} = \frac{\lambda}{\psi_0^2}\varepsilon'.$$

Stability of the power protection property

It is more complex to show that the assumption of power protection is preserved under metric perturbation. Previously, we have only considered two similar but arbitrary Riemannian metric fields g and g' on a neighborhood U. We now restrict ourselves to the case where g' is a uniform metric field. We shall always compare the metric field g in a neighborhood U with the uniform metric field $g' = g_0 = g(p_0)$ where $p_0 \in U$. Because g_0 and the Euclidean metric field differ only by a linear transformation, we simplify matters and assume that g_0 is the Euclidean metric field.

We now give conditions such that the point set is also protected with respect to g_0. A few intermediary steps are needed to prove the main result:

- Given two seeds, we prove that the bisectors of these two seeds in the Voronoi diagrams built with respect to g and with respect to $g' = g_{\mathbb{E}}$ are close (Lemma 3.4.5).

- We prove that the Voronoi cell of a point p_0 with respect to g, $\mathrm{V}_g(p_0)$ can be encompassed by two scaled versions (one larger and one smaller) of the Euclidean Voronoi cell $\mathrm{V}_{\mathbb{E}}(p_0)$ (Lemma 3.4.12).
- We combine this encompassing with bounds on the dihedral angles of Euclidean Delaunay simplices given by Lemma 3.3.7 to compute a stability region around Voronoi vertices where the same combinatorial Voronoi vertex of lives for both $\mathrm{V}_g(p_0)$ and $\mathrm{V}_{\mathbb{E}}(p_0)$ in 2D (Lemma 3.4.13). We then extend it to any dimension by induction (Lemma 3.4.14).

The main result appears in Lemma 3.4.16 and gives the stability of power protection under metric perturbation.

We first define the scaled version of a Voronoi cell more rigorously.

Definition 3.4.4 (Relaxed Voronoi cell) *Let $\omega \in \mathbb{R}$. The relaxed Voronoi cell of the seed p_0 is*

$$\mathrm{V}_g^{+\omega}(p_0) = \{x \in U \mid d_g(p_0, x)^2 \leq d_g(p_i, x)^2 + \omega \text{ for all } i \neq 0\}.$$

The following lemma expresses that two Voronoi cells computed in similar metric fields are close.

Lemma 3.4.5 *Let $U \subset \Omega$ be open, and g and g' be two Riemannian metric fields on U. Let $\psi_0 \geq 1$ be a bound on the metric distortion. Suppose that U is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$, such that $\forall p \in B_g(p_0, r_0), \psi(g(p), g'(p)) \leq \psi_0$. Let $\mathcal{P}_U = \{p_i\}$ be a finite point set in U. Let $\mathrm{V}_{p_0,g}$ denote a Voronoi cell with respect to the Riemannian metric field g.*

Suppose that the Voronoi cell $\mathrm{V}_{g'}^{2\rho^2(\psi_0^2-1)}(p_0)$ lies in a ball of radius ρ with respect to the metric g', which lies completely in U. Let $\omega_0 = 2\rho^2(\psi_0^2-1)$. Then $\mathrm{V}_g(p_0)$ lies in $\mathrm{V}_{g'}^{+\omega_0}(p_0)$ and contains $\mathrm{V}_{g'}^{-\omega_0}(p_0)$.

Proof. Let $\mathrm{BS}_g(p_0, p_i)$ be the bisector between p_0 and p_i with respect to g. Let $y \in \mathrm{BS}_g(p_0, p_i) \cap B_{g'}(p_0, \rho)$, where $B_{g'}(p_0, \rho)$ denotes the ball centered at p_0 of radius ρ with respect to g'. Now $d_g(y, p_0) = d_g(y, p_i)$, and thus

$$\begin{aligned}
|d_{g'}(y,p_0)^2 - d_{g'}(y,p_i)^2| &= \left|d_{g'}(y,p_0)^2 - d_g(y,p_0)^2 + d_g(y,p_i)^2 - d_{g'}(y,p_i)^2\right| \\
&\leq \left|d_{g'}(y,p_0)^2 - d_g(y,p_0)^2\right| + \left|d_{g'}(y,p_i)^2 - d_g(y,p_i)^2\right| \\
&\leq (\psi_0^2 - 1)(d_{g'}(y,p_0)^2 + d_{g'}(y,p_i))^2 \\
&\leq 2\rho^2(\psi_0^2 - 1).
\end{aligned}$$

Thus $d_{g'}(y,p_0)^2 \leq d_{g'}(y,p_i)^2 + \omega$ and $d_{g'}(y,p_0)^2 \geq d_{g'}(y,p_i)^2 - \omega$ with $\omega = 2\rho^2(\psi_0^2 - 1)$, which gives us the expected result. □

Remark 3.4.6 *Lemma 3.4.5 does not require g' to be a uniform metric field.*

We clarify in the next lemma that the bisectors of a Voronoi diagram with respect to a uniform are affine hyperplanes.

Lemma 3.4.7 *Let $U \subset \Omega$ be open, and g and g' be two Riemannian metric fields on U. Let $\psi_0 \geq 1$ be a bound on the metric distortion. Suppose that U is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$, such that $\forall p \in B(p_0, r_0), \psi(g(p), g'(p)) \leq \psi_0$. Let $\mathcal{P}_U = \{p_i\}$ be a finite point set in U. Let g' be a uniform metric field. We refer to g' as g_0 to emphasis its constancy. Let $p_0 \in \mathcal{P}_U$. The bisectors of $\mathrm{V}_{g_0}^{\pm\omega_0}(p_0)$ are hyperplanes.*

Proof. The bisectors of $\mathrm{V}_{g_0}^{\pm\omega_0}(p_0)$ are given by

$$\mathrm{BS}_{g_0}^{\pm\omega_0}(p_0,p_i)=\left\{x\in\Omega \mid d_{g_0}(p_0,x)^2=d_{g_0}(p_i,x)^2\pm\omega_0\right\}.$$

For $x\in\mathrm{BS}_{g_0}^{\pm\omega_0}(p_0,p_i)$, we have by definition that

$$\begin{aligned}
&\|x-p_0\|_{g_0}^2=\|x-p_i\|_{g_0}^2\pm\omega_0\\
\iff\ &(x-p_0)^t g_0(x-p_0)=(x-p_i)^t g_0(x-p_i)\pm\omega_0\\
\iff\ &2x^t g_0(p_i-p_0)=p_i^t g_0 p_i-p_0^t g_0 p_0\pm\omega_0.
\end{aligned}$$

which is the equation of an hyperplane since g_0 is uniform. □

The cells $\mathrm{V}_{g_0}^{\pm\omega_0}(p_0)$ are unfortunately impractical to manipulate as we do not have an explicit distance between the boundaries $\partial\,\mathrm{V}_{g_0}(p_0)$ and $\partial\,\mathrm{V}_{g_0}^{\pm\omega_0}(p_0)$. However that distance can be bounded; this is the purpose of the following lemma.

Lemma 3.4.8 *Let $U\subset\Omega$ be open, and g and g' be two Riemannian metric fields on U. Let $\psi_0\geq 1$ be a bound on the metric distortion. Suppose that U is included in a ball $B_g(p_0,r_0)$, with $p_0\in U$ and $r_0\in\mathbb{R}^+$, such that $\forall p\in B(p_0,r_0),\psi(g(p),g'(p))\leq\psi_0$. Let $\mathcal{P}_U=\{p_i\}$ be a finite point set in U. Assume furthermore that g_0 is the Euclidean metric field. Let $p_0\in\mathcal{P}_U$. We have*

$$d_{g_0}(\partial\,\mathrm{V}_{g_0}(p_0),\partial\,\mathrm{V}_{g_0}^{\pm\omega_0}(p_0)) := \min_{\substack{x\in\partial\,\mathrm{V}_{g_0}(p_0)\\ y\in\partial\,\mathrm{V}_{g_0}^{\pm\omega_0}(p_0)}} d_{g_0}(x,y)\leq\frac{\rho_0^2(\psi_0^2-1)}{\mu_0},$$

with ρ_0 defined as ρ is in Lemma 3.4.5.

Proof. Let m_{ω_0} be the intersection of the segment $[p_0,p_i]$ and the bisector $\mathrm{BS}_{g_0}^{-\omega_0}(p_0,p_i)$, for $i\neq 0$. Let m be the intersection of the segment $[p_0,p_i]$ and $\partial\,\mathrm{V}_{g_0}(p_0)$, for $i\neq 0$. We have

$$\begin{cases} m_{\omega_0}\in\mathrm{BS}_{g_0}^{-\omega_0}(p_0,p_i) & \iff 2m_{\omega_0}^T(p_i-p_0)=p_i^Tp_i-p_0^Tp_0-\omega_0\\ m\in\partial\,\mathrm{V}_{g_0}(p_0) & \iff 2m^T(p_i-p_0)=p_i^Tp_i-p_0^Tp_0\end{cases}$$

Therefore

$$2(m-m_{\omega_0})^T(p_i-p_0)=\omega_0.$$

Since $(m-m_{\omega_0})$ and (p_i-p_0) are linearly dependent,

$$\omega_0=2\,|m-m_{\omega_0}|\,|p_i-p_0|\,.$$

$\mathcal{P}$ is μ_0-separated, which implies that

$$\omega_0\geq 2\,|m-m_{\omega_0}|\,\mu_0,$$

and

$$|m-m_{\omega_0}|\leq\frac{\omega_0}{2\mu_0}=\frac{\rho_0^2(\psi_0^2-1)}{\mu_0}.$$

□

Definition 3.4.9 *In the following, we use* $\rho = 2\varepsilon_0$*, and therefore* $\omega_0 = 8\varepsilon_0^2(\psi_0^2 - 1)$*. We show that this choice is reasonable in Lemma 3.4.15. Additionally, we define*

$$\eta_0 = \frac{\rho_0^2(\psi_0^2 - 1)}{\mu_0}.$$

We are now ready to encompass the Riemannian Voronoi cell of p_0 with respect to an arbitrary metric field g with two scaled versions of the Euclidean Voronoi cell of p_0. The notions of *dilated* and *eroded* Voronoi cells will serve the purpose of defining precisely these scaled cells.

Definition 3.4.10 (Eroded Voronoi cell) *Let* $\omega \in \mathbb{R}$*. The eroded Voronoi cell of* p_0 *is the morphological erosion of* $\mathrm{V}_g(p_0)$ *by a ball of radius* ω*:*

$$\mathrm{EV}_g^{-\omega}(p_0) = \{x \in \mathrm{V}_g(p_0) \mid d_g(x, \partial\,\mathrm{V}_g(p_0)) > \omega\}.$$

Definition 3.4.11 (Dilated Voronoi cell) *Let* $\omega \in \mathbb{R}$*. The dilated Voronoi cell of* p_0 *is:*

$$\mathrm{DV}_g^{+\omega}(p_0) = \bigcap_{i \neq 0} H^{\omega}(p_0, p_i),$$

where $H^{\omega}(p_0, p_i)$ *is the half-space containing* p_0 *and delimited by the bisector* $\mathcal{BS}(p_0, p_i)$ *translated away from* p_0 *by* ω_0 *(see Figure 3.8).*

The second important step on our path towards the stability of power protection is the encompassing of the Riemannian Voronoi cell, and is detailed below.

Lemma 3.4.12 *Let* $U \subset \Omega$ *be open, and* g *and* g' *be two Riemannian metric fields on* U*. Let* $\psi_0 \geq 1$ *be a bound on the metric distortion. Suppose that* U *is included in a ball* $B_g(p_0, r_0)$*, with* $p_0 \in U$ *and* $r_0 \in \mathbb{R}^+$*, such that* $\forall p \in B(p_0, r_0), \psi(g(p), g'(p)) \leq \psi_0$*. Let* $\mathcal{P}_U = \{p_i\}$ *be a finite point set in* U*. We have*

$$\mathrm{EV}_{g_0}^{-\eta_0}(p_0) \subseteq \mathrm{V}_{g_0}^{-\omega_0}(p_0) \subseteq \mathrm{V}_{g_0}(p_0) \subseteq \mathrm{V}_{g_0}^{+\omega_0}(p_0) \subseteq \mathrm{DV}_{g_0}^{+\eta_0}(p_0).$$

These inclusions are illustrated in Figure 3.8.

Proof. Using the notations and the result of Lemma 3.4.8, we have

$$|m - m_{\omega_0}| \leq \eta_0.$$

Since the bisectors $\mathrm{BS}_{g'}^{\omega_0}(p_0, p_i)$ are hyperplanes, the result follows. □

In Figures 3.8 and 3.9, $\mathrm{DV}_{g_0}^{+\eta_0}$ and $\mathrm{EV}_{g_0}^{-\eta_0}$ are shown in green and V_{g_0} in yellow.

The next step is to prove that we have stability of the Voronoi vertices of $\mathrm{V}_g(p_0)$, meaning that a small perturbation of the metric only creates a small displacement of any Voronoi vertex of the Voronoi cell of p_0. The following lemma shows that Voronoi vertices are close if the distortion between the metric fields g and g_0 is small. The approach is to use Lemmas 3.4.5 and 3.4.12: we know that each $(n-1)$-face of the Riemannian Voronoi cell $\mathrm{V}_g(p_0)$ and shared by a Voronoi cell $V_g(q)$ is enclosed within the bisectors of $\mathrm{DV}_{g_0}(p_0)$ and $\mathrm{EV}_{g_0}(p_0)$ (which are translations of the bisectors of $\mathrm{V}_{g_0}(p_0)$) for the seeds p_0 and q. These two bisectors create a "band" that contains the bisector $\mathrm{BS}_g(p_0, q)$. Given a Voronoi vertex c in $\mathrm{V}_g(p_0)$, c can be obtained as the intersection of $n+1$ Voronoi cells, but also as the intersection of n Voronoi $(n-1)$-faces of $\mathrm{V}_g(p_0)$. The intersection of the bands associated to those n $(n-1)-$faces is a parallelotopic-shaped region which by definition contains the same (combinatorially speaking) Voronoi vertex, but for $\mathrm{V}_{g_0}(p_0)$. Lemmas 3.4.13 and 3.4.14 express this reasoning, which is illustrated in Figure 3.9 for 2D and 3.10 for any dimension.

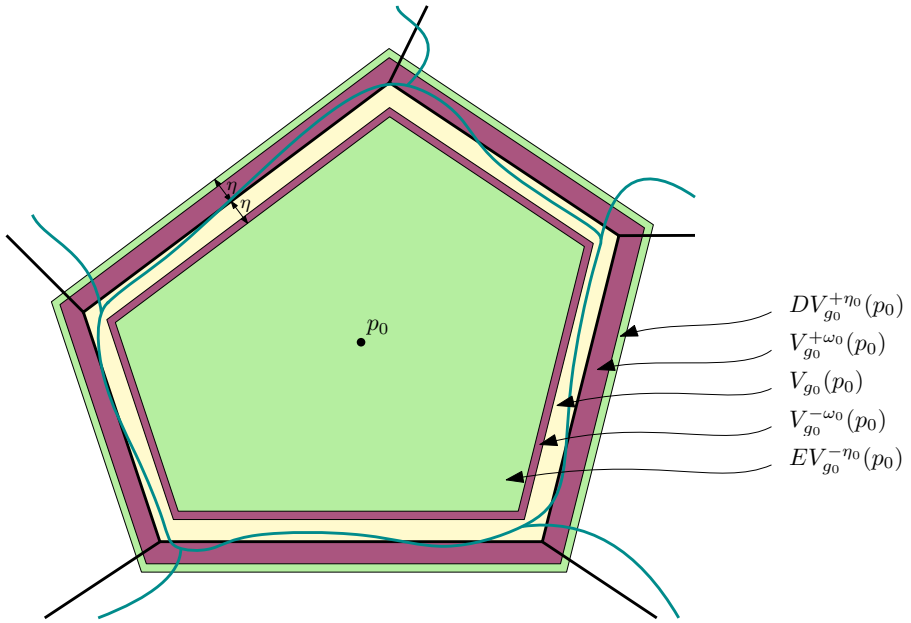

Figure 3.8: The different encompassing cells around p_0. The Riemannian Voronoi diagrams with respect to g and g_0 are traced in dark cyan and black respectively. The Voronoi cell $\mathrm{V}_{g_0}(p_0)$ is colored in yellow. The cells $\mathrm{DV}_{g_0}^{+\eta_0}$ and $\mathrm{EV}_{g_0}^{-\eta_0}$ are colored in green, and the cells $\mathrm{V}_{g_0}^{\pm\omega_0}(p_0)$ are colored in purple.

Lemma 3.4.13 *We consider here $\Omega = \mathbb{R}^2$. Let $U \subset \Omega$ be open, and g and g' be two Riemannian metric fields on U. Let $\psi_0 \geq 1$ be a bound on the metric distortion. Suppose that U is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$, such that $\forall p \in B(p_0, r_0), \psi(g(p), g'(p)) \leq \psi_0$. Let $\mathcal{P}_U = \{p_i\}$ be a finite point set in U. Assume furthermore that $\mathcal{P}_U$ is a δ_0-power protected (ε_0, μ_0)-net with respect to g_0 (the Euclidean metric). Let $p_0 \in \mathcal{P}_U$. Let c and c_0 be the same Voronoi vertex in respectively $\mathrm{V}_g(p_0)$ and $\mathrm{V}_{g_0}(p_0)$. Then $d_{g_0}(c, c_0) \leq \chi_2$ with*

$$\chi_2 = \frac{2\eta_0}{\sqrt{1 + \frac{\mu_0}{2\varepsilon_0}} - \sqrt{1 - \frac{\mu_0}{2\varepsilon_0}}} = \frac{2\eta_0}{\sqrt{1 + \frac{\lambda}{2\psi_0^2}} - \sqrt{1 - \frac{\lambda}{2\psi_0^2}}}.$$

where λ is given by $\mu_0 = \frac{\lambda}{\psi_0^2}\varepsilon_0$ (see Lemma 3.4.3).

Proof. We use Lemma 3.4.12. $\mathrm{V}_g(p_0)$ lies in $\mathrm{DV}_{g_0}^{+\eta_0}$ and contains $\mathrm{EV}_{g_0}^{-\eta_0}$. The circumcenters c and c_0 lie in a parallelogrammatic region centered on c_0, itself included in the ball centered on c_0 and with radius χ. The radius χ is given by half the length of the longest diagonal of the parallelogram (see Figure 3.9). By Lemma 3.4.2, $\mathcal{P}$ is an (ε_0, μ_0)-net with respect to g_0. Let θ be the angle of the Voronoi corner of $\mathrm{V}_{g_0}(p_0)$ at c_0. By Lemma 3.3.1, that angle is bounded:

$$\theta_m = 2\arcsin\left(\frac{\mu_0}{2\varepsilon_0}\right) \leq \theta \leq \pi - \arcsin\left(\frac{\mu_0}{2\varepsilon_0}\right) = \theta_M.$$

Since $\pi - \theta_M < \theta_m$, χ is maximal when $\theta > \pi/2$. We thus assume $\theta > \pi/2$, and compute a bound on

χ as follows:

$$\begin{aligned}\sin\left(\frac{\pi-\theta}{2}\right) = \frac{\eta_0}{\chi} \Longrightarrow \chi &= \frac{\eta_0}{\sin\left(\frac{\pi-\theta}{2}\right)} \\ &\leq \frac{\eta_0}{\sin\left(\frac{1}{2}\arcsin\left(\frac{\mu_0}{2\varepsilon_0}\right)\right)} \\ &\leq \frac{2\eta_0}{\sqrt{1+\frac{\mu_0}{2\varepsilon_0}}-\sqrt{1-\frac{\mu_0}{2\varepsilon_0}}} =: \chi_2,\end{aligned}$$

using $\sin(\frac{1}{2}\arcsin(\alpha)) = \frac{1}{2}\left(\sqrt{1+\alpha}-\sqrt{1-\alpha}\right)$.

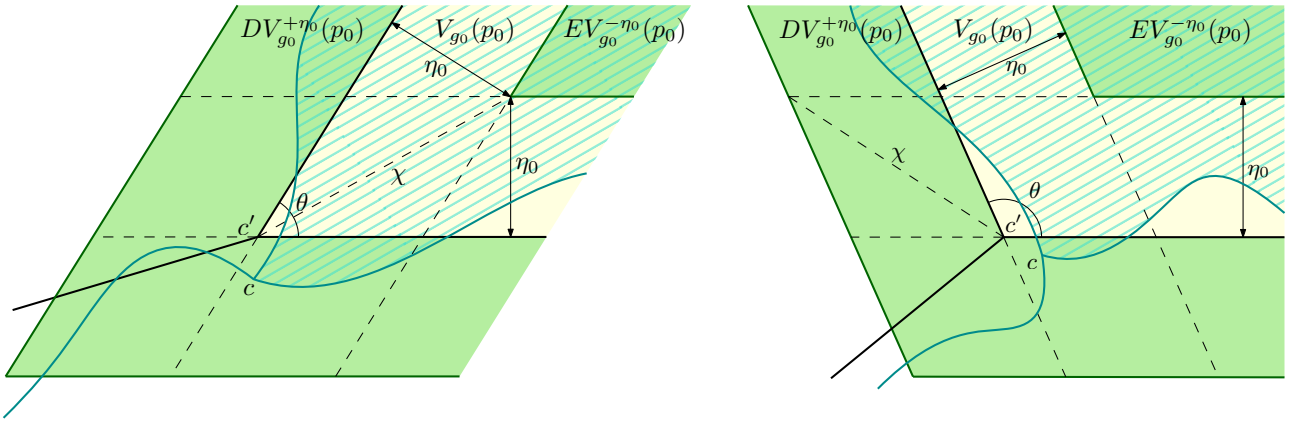

Figure 3.9: Black lines trace Vor_{g_0} and cyan lines trace Vor_g. The cell $\mathrm{V}_{g_0}(p_0)$ is colored in yellow and the cell $\mathrm{V}_g(p_0)$ is dashed. The green regions correspond to $\mathrm{DV}_{g_0}^{+\eta}(p_0)$ and $\mathrm{EV}_{g_0}^{-\eta}(p_0)$. On the left, the configuration where $\theta < \pi/2$; on the right, $\theta > \pi/2$.

□

The result obtained in Lemma 3.4.13 can be extended to any dimension using induction and the stability of the Voronoi vertices of facets.

Lemma 3.4.14 *We consider here $\Omega = \mathbb{R}^n$. Let $U \subset \Omega$ be open, and g and g' be two Riemannian metric fields on U. Let $\psi_0 \geq 1$ be a bound on the metric distortion. Suppose that U is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$, such that $\forall p \in B(p_0, r_0), \psi(g(p), g'(p)) \leq \psi_0$. Let $\mathcal{P}_U = \{p_i\}$ be a δ_0-power protected (ε_0, μ_0)-net with respect to g_0 (the Euclidean metric). Let $p_0 \in \mathcal{P}_U$. Let c and c_0 be the same Voronoi vertex in respectively $\mathrm{V}_g(p_0)$ and $\mathrm{V}_{g_0}(p_0)$. Then $d_{g_0}(c, c_0) \leq \chi$ with*

$$\chi = \frac{\chi_2}{\sin^{n-2}\left(\frac{\varphi_0}{2}\right)}.$$

where χ_2 is defined as in Lemma 3.4.13, and φ_0 is the maximal dihedral angle between two faces of a simplex.

Proof. We know from Lemma 3.4.5 that $\mathrm{V}_g(p_0)$ lies in $\mathrm{DV}_{g_0}^{+\eta_0}$ and contains $\mathrm{EV}_{g_0}^{-\eta_0}$. The circumcenters c and c_0 lie in a parallelotopic region centered on c_0 defined by the intersection of n Euclidean thickened Voronoi faces. This parallelotope and its circumscribing sphere are difficult to compute. However, it can be seen as the intersection of two parallelotopic tubes defined by the intersection of

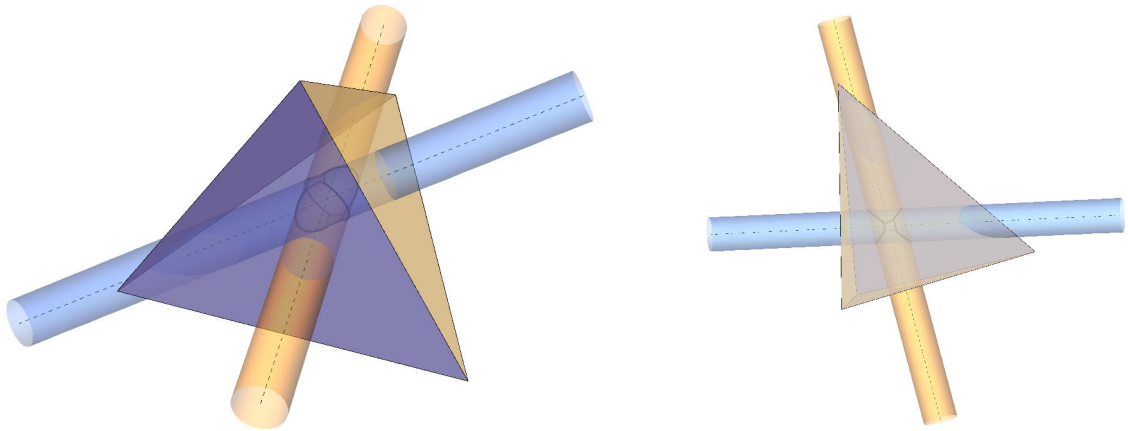

Figure 3.10: Left, a simplex, the duals of two faces and their respective thickened duals (cylinders); the orange thickened dual is orthogonal to the purple face (and inversely). Right, the intersection of the two tubes, seen from above, illustrates the proof of Lemma 3.4.14.

$n-1$ Euclidean thickened Voronoi faces. From another point of view, this is the computation of the intersection of the thickened duals of two facets τ_1 and τ_2 incident to p_0 of the simplex $\sigma \in \mathrm{Del}_{g_0}(\mathcal{P})$, dual of c (and c_0), see Figure 3.10 (left).

The stability radius χ is computed incrementally by increasing the dimension and proving stability of the circumcenters of the faces of the simplices. We prove the formula by induction.

The radius of each tube is given by the stability of the radius of the circumcenter in the lower dimension of the facet. The base case, $\mathbb{R}^n = \mathbb{R}^3$, is solved by Lemma 3.4.13, and gives

$$\chi_2 = \frac{2\eta_0}{\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}}.$$

We now consider two facets τ_1 and τ_2 that are incident to p_0. Denote $\mathcal{D}_1$ and $\mathcal{D}_2$ their respective duals. By Lemma 3.4.5, $\mathcal{D}_1$ and $\mathcal{D}_2$ lie in two cylinders C_1 and C_2 of radius χ_2. C_1 and C_2 are also orthogonal to τ_1 and τ_2 and c and c_0 lie in $C_1 \cap C_2$. The angle φ between C_1 and C_2 is exactly the dihedral angle between τ_1 and τ_2. By Lemma 3.3.7, we have

$$\arcsin(s_0) \leq \varphi \leq \pi - \arcsin(s_0) \text{ with } s_0 = \frac{1}{2}\left[\frac{\iota^2}{4\psi_0^2} - \frac{1}{2}\left(\psi_0^2 - \frac{1}{\psi_0^2}\right)\right]$$

Let $\varphi_0 = \arcsin(s_0) = \pi - \arcsin(s_0)$. We encompass the intersection of the cylinders, difficult to compute, with a sphere whose radius can be computed as follows (see Figure 3.10, right):

$$\chi_3 = \frac{\chi_2}{\cos(\alpha)} \text{ with } \alpha = \frac{\pi}{2} - \frac{\varphi_0}{2}.$$

Thus,

$$\chi_3 = \frac{\chi_2}{\sin\left(\frac{\varphi_0}{2}\right)}.$$

Recursively,

$$\chi = \frac{\chi_2}{\sin^{n-2}\left(\frac{\varphi_0}{2}\right)}.$$

□

We have assumed in different lemmas that we could pick values of ρ_0 or ω_0 that fit our need. The following lemma shows that these assumptions were reasonable.

Lemma 3.4.15 *Lemmas 3.4.12 and 3.4.14 allow us to characterize the parameter ρ_0 more precisely. Indeed, an assumption of Lemma 3.4.5 was that $\mathrm{V}_{g_0}^{\omega_0}(p_0)$ is included in a ball $B_{g_0}(p_0, \rho_0)$. If the sampling of $\mathcal{P}$ is sufficiently dense, such an assumption is reasonable.*

Proof. By definition, the Voronoi cell $\mathrm{V}_{g_0}^{\omega_0}(p_0)$ is included in the dilated cell $\mathrm{DV}_{g_0}^{+\eta_0}(p_0)$. Since the point set is an ε-sample, we have $d_{g_0}(p_0, x) \leq \varepsilon_0$ for $x \in \mathrm{V}_{g_0}(p_0)$. By Lemma 3.4.14, we have for $x \in \mathrm{DV}_{g_0}^{+\eta_0}(p_0)$

$$d_{g_0}(p_0, x) \leq \varepsilon_0 + \chi.$$

Recall from Lemma 3.4.14 that

$$\chi = \frac{\chi_2}{\left[\sin\left(\frac{1}{2}\arcsin\left(s_0\right)\right)\right]^{n-2}}.$$

with $\eta_0 = \frac{\rho_0^2(\psi_0^2-1)}{\mu_0}$ and $s_0 = \frac{1}{2}\left[\frac{\iota^2}{4\psi_0^2} - \frac{1}{2}\left(\psi_0^2 - \frac{1}{\psi_0^2}\right)\right]$. We require $\mathrm{V}_{g_0}^{\omega_0}(p_0) \subset B_{g_0}(p_0, \rho_0)$, which is verified if $\mathrm{DV}_{g_0}^{+\eta_0}(p_0) \subset B_{g_0}(p_0, \rho_0)$, that is if

$$\begin{aligned}
&\varepsilon_0 + \frac{\chi_2}{\left[\sin\left(\frac{1}{2}\arcsin\left(s_0\right)\right)\right]^{n-2}} \leq \rho_0 \\
\iff &\frac{4\eta_0}{\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0} - \sqrt{1-s_0}\right)^{n-2}} \leq \rho_0 - \varepsilon_0 \\
\iff &\frac{4\rho_0^2(\psi_0^2-1)}{\mu_0\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0} - \sqrt{1-s_0}\right)^{n-2}} \leq \rho_0 - \varepsilon_0 \\
\iff &\frac{4(\psi_0^2-1)}{\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0} - \sqrt{1-s_0}\right)^{n-2}} \leq \frac{\mu_0(\rho_0-\varepsilon_0)}{\rho_0^2} \\
\iff &\frac{4\psi_0(\psi_0^2-1)}{\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0} - \sqrt{1-s_0}\right)^{n-2}} \leq \frac{\lambda\varepsilon(\rho_0-\varepsilon_0)}{\rho_0^2}, \text{ by Remark 3.4.3.}
\end{aligned}$$

The parameter ρ_0 can be chosen arbitrarily as long as it is greater than ε_0, and we have taken $\rho_0 = 2\varepsilon_0$ (see Definition 3.4.9), which imposes

$$\frac{\psi_0^2(\psi_0^2-1)}{\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0} - \sqrt{1-s_0}\right)^{n-2}} \leq \frac{\lambda}{16}. \tag{3.7}$$

Recall that the parameter λ is fixed. By continuity of the metric field, $\lim_{\varepsilon\to 0}\psi_0 = 1$, therefore the left hand side goes to 0 and Inequality (3.7) is eventually satisfied as the sampling is made denser. □

Finally, we can now show the main result: the power protection property is preserved when the metric field is perturbed.

Lemma 3.4.16 *Let $U \subset \Omega$ be open, and g and g' be two Riemannian metric fields on U. Let $\psi_0 \geq 1$ be a bound on the metric distortion. Suppose that U is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$, such that $\forall p \in B(p_0, r_0), \psi(g(p), g'(p)) \leq \psi_0$. Assume that $\mathcal{P}_U$ is a δ-power protected*

(ε, μ)-net in U with respect to g. If δ is well chosen, then $\mathcal{P}_U$ is a δ_0-power protected net with respect to g_0, with

$$\delta_0^2 = \left(\frac{1}{\psi_0^2} - \psi_0^2\right)(\varepsilon + \chi)^2 - \frac{4\varepsilon\chi}{\psi_0^2} + \frac{\delta^2}{\psi_0^2}.$$

Proof. By Lemma 3.4.2, we know that $\mathcal{P}_U$ is (ε_0, μ_0)-net with respect to g_0. Let $q \in \mathcal{P}_U$, with q not a vertex in the dual of c. Let c_0 be the combinatorial equivalent of c in $\mathrm{V}_{g_0}(\mathcal{P})$ Since $\mathcal{P}$ is a δ-power protected net with respect to g, we have $d_g(c, q) > \sqrt{r^2 + \delta^2}$, where $r = d_g(c, p)$. On the one hand, we have

$$\begin{aligned} d_{g_0}(c_0, q) &\geq d_{g_0}(q, c) - d_{g_0}(c, c_0) \\ &\geq \frac{1}{\psi_0} d_g(q, c) - \chi \\ &\geq \frac{1}{\psi_0}\sqrt{r^2 + \delta^2} - \chi. \end{aligned}$$

by Lemma 3.4.13. On the other hand, for any $p \in \mathcal{P}_U$ such that p is a vertex of the dual of c, we have

$$\begin{aligned} r_0 = d_{g_0}(c_0, p) &\leq d_{g_0}(c, p) + d_{g_0}(c_0, c) \\ &\leq \psi_0 d_g(c, p) + \chi \\ &\leq \psi_0 r + \chi. \end{aligned}$$

Thus δ_0-power protection of $\mathcal{P}_U$ with respect to g_0 requires

$$\begin{aligned} &\frac{1}{\psi_0}\sqrt{r^2 + \delta^2} - \chi > \chi + \psi_0 r \\ \iff &\sqrt{r^2 + \delta^2} > \psi_0(2\chi + \psi_0 r). \end{aligned}$$

This is verified if

$$\begin{aligned} \delta^2 > \left(\psi_0(2\chi + \psi_0 r)\right)^2 - r^2 &= 4\chi^2\psi_0^2 + 4\chi\psi_0^3 r + \psi_0^4 r^2 - r^2 \\ &= 4\chi^2\psi_0^2 + 4\chi\psi_0^3 r + (\psi_0^4 - 1)r^2, \end{aligned}$$

for all $r \in [\mu/2, \varepsilon]$. This gives us

$$\delta^2 > 4\chi^2\psi_0^2 + 4\chi\psi_0^3\varepsilon + (\psi_0^4 - 1)\varepsilon^2. \tag{3.8}$$

This condition on δ is only reasonable if the right hand side is not too large. Indeed, since $\mathcal{P}$ is an ε-sample, we must have $d_g(c, q) < 2\varepsilon$. However, we have that $d_g(c, q)^2 > d_g(c, p)^2 + \delta^2$ by δ-power protection of $\mathcal{P}$ with respect to g. Because $d_g(c, p) < \varepsilon$, it suffices that $\delta < \varepsilon$. We will now show that this is reasonable by examining the limit of the right hand side of Inequality (3.8).

We note, see Lemma 3.4.14, that

$$\chi = \frac{\chi_2}{\sin^{n-2}\left(\frac{\varphi}{2}\right)} = \frac{4\eta}{\left(\sqrt{1 + \frac{\lambda}{2\psi_0^2}} - \sqrt{1 - \frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1 + s_0} - \sqrt{1 - s_0}\right)^{n-2}},$$

where $\varepsilon_0 = \psi_0\varepsilon$ and $\mu_0 = \mu/\psi_0 = \lambda\varepsilon/\psi_0$ (see Remark 3.4.3). So that

$$\begin{aligned}4\chi^2\psi_0^2 + 4\chi\psi_0^3\varepsilon + (\psi_0^4 - 1)\varepsilon^2 = 4&\left(\frac{16\varepsilon\psi_0^4(\psi_0^2-1)}{\lambda\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0}-\sqrt{1-s_0}\right)^{n-2}}\right)^2 \\ &+ 4\frac{16\varepsilon\psi_0^3(\psi_0^2-1)}{\lambda\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0}-\sqrt{1-s_0}\right)^{n-2}}\psi_0^3 \\ &+ (\psi_0^2-1)(\psi_0^2+1)\varepsilon^2. \end{aligned} \tag{3.9}$$

This means that the right hand side of (3.8) is of the form $f(\psi_0)(\psi_0^2-1)\varepsilon^2$, where $f(\psi_0)$ is a function that tends to a constant as ψ_0 goes to 1:

$$\begin{aligned} f(\psi_0) = 4&\left(\frac{16\psi_0^4}{\lambda\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0}-\sqrt{1-s_0}\right)^{n-2}}\right)^2 \\ &+ 4\frac{16\psi_0^6}{\lambda\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0}-\sqrt{1-s_0}\right)^{n-2}} \\ &+ (\psi_0^2+1) \\ \xrightarrow{\psi_0\to 1} 4&\left(\frac{16}{\lambda\left(\sqrt{1+\frac{\lambda}{2}} - \sqrt{1-\frac{\lambda}{2}}\right)\left(\sqrt{1+\frac{\iota^2}{4}}-\sqrt{1-\frac{\iota^2}{4}}\right)^{n-2}}\right)^2 \\ &+ 4\frac{16}{\lambda\left(\sqrt{1+\frac{\lambda}{2}} - \sqrt{1-\frac{\lambda}{2}}\right)\left(\sqrt{1+\frac{\iota^2}{4}}-\sqrt{1-\frac{\iota^2}{4}}\right)^{n-2}} \\ &+ 2 \end{aligned}$$

So the bound given in Equality (3.8) may be easily satisfied if the metric distortion is sufficiently small.

We now provide an explicit value for δ_0 in terms of δ. Let $\xi = d_g(c,q)$ and $\xi_0 = d_{g_0}(c_0,q_0)$. We have the following bounds on r_0 and ξ_0:

$$\begin{aligned}\frac{1}{\psi_0}(r-\xi) \le r_0 \le \psi_0(r+\xi) \\ \frac{1}{\psi_0}\sqrt{(r-\chi)^2+\delta^2} \le \xi_0 \le \psi_0\sqrt{(r+\chi)^2+\delta^2}.\end{aligned}$$

If we had $\tilde{\delta}$-power protection, we would have

$$\begin{aligned} r_0^2 + \tilde{\delta}^2 \le \xi_0^2 &\iff \tilde{\delta}^2 \le \xi_0^2 - r_0^2 \\ &\iff \tilde{\delta}^2 \le \frac{1}{\psi_0^2}\left((r-\chi)^2+\delta^2\right) - \psi_0^2(r+\chi)^2 \\ &\iff \tilde{\delta}^2 \le \frac{1}{\psi_0^2}(r+\chi)^2 - \frac{4r\chi}{\psi_0^2} + \frac{\delta^2}{\psi_0^2} - \psi_0^2(r+\chi)^2 \\ &\impliedby \tilde{\delta}^2 \le \left(\frac{1}{\psi_0^2} - \psi_0^2\right)(\varepsilon+\chi)^2 - \frac{4\varepsilon\chi}{\psi_0^2} + \frac{\delta^2}{\psi_0^2}. \end{aligned}$$

Therefore we can take $\delta_0^2 = \frac{\delta^2}{\psi_0^2} + \left(\frac{1}{\psi_0^2} - \psi_0^2\right)(\varepsilon + \chi)^2 - \frac{4\varepsilon\chi}{\psi_0^2}$. Note that with this definition, δ_0 goes to δ as ψ_0 goes to 1, which proves that our value of δ_0 is legitimate. □

3.5 Conclusion

We have proved in this chapter various results on the quality of Delaunay triangulations and Voronoi diagrams built from specific types of point sets, power protected nets. These results form the basis of all the theoretical work on the algorithms presented in other chapters of this thesis.

Part I

Anisotropic Delaunay triangulations

4. Locally uniform anisotropic meshes

The Delaunay triangulation has been extensively studied and is well known to possess useful and well-defined properties (see Preparata and Shamos [117]). Attempts have been made to extend the notion of Delaunay triangulation to the anisotropic setting either by adapting the famous Bowyer-Watson algorithm (Mavriplis [102], Borouchaki [33], Dobrzynski [60], Alauzet [4]) or as the dual of an anisotropic Voronoi diagram (see Labelle and Shewchuk [89], Du and Wang [64]). However, neither class of method offer in all dimensions the theoretical guarantees nor the practical robustness that we are interested in.

A theoretically sound framework for Delaunay-based anisotropic meshes was introduced by Boissonnat et al. [30], called "locally uniform anisotropic meshes". Locally uniform anisotropic meshes are simplicial complexes in which the star of each vertex is Delaunay for the metric attached to the vertex. The use of anisotropic stars, inspired by the works of Shewchuk [128] and Schoen [126], is combined with the sliver removal techniques proposed by Li and Teng [94, 96]. These techniques are adapted to the anisotropic setting to construct a star-based refinement algorithm that cleverly selects refinement points. The algorithm works in any dimension and offers guarantees on its termination and on the quality (size and shape) of the final simplices.

While the theoretical aspect of the approach has been thoroughly studied [30, 31], its practicality is comparatively not as well explored. This chapter intends to fill this gap and presents a comprehensive empirical study of the algorithm. We first recall the required theoretical basis and detail some minor changes brought to the theory after practical experimentation. We then dive in the heuristic analysis of the behavior of the algorithm and its parameters and propose some improvements. A strong focus is put on the phenomenon of inconsistencies, which turns out to be a real issue of the algorithm.

Contributions We introduce an implementation of the anisotropic Delaunay refinement algorithm introduced by Boissonnat et al. [30]. We specifically detail the technical issues, and their solution, that arise when aiming for efficiency. This implementation of the star set is completely new and results in significant improvements in terms of computational speed, robustness and genericity over the former implementation that was used in [28].

The empirical study of our implementation and of the algorithm is comprehensive and produce (minor) theoretical improvements. We investigate the practicality of the algorithm, analyze its limitations and propose various fixes that attempt to remedy these limitations.

Contents

4.1 The star set

Many algorithms have been devised to construct Euclidean Delaunay triangulations, but they cannot be simply extended to arbitrary metric fields. Although the computation of a curved Riemannian Delaunay triangulation is difficult, it is easy to compute the Delaunay triangulation of a set of points $\mathcal{P}$ with respect to the metric G_p ($\mathrm{Del}_p(\mathcal{P})$). Indeed, recall from Chapter 2 that

$$d_p(a,b) = \sqrt{(a-b)^t F_p^t F_p (a-b)} = \|F_p(a-b)\|\,,$$

where F_p is the square root of G_p. Any metric-dependent geometric construction on a set of points $\mathcal{P}$, like the Voronoi diagram or a simplex circumcircle, can therefore be obtained for the metric G_p by the following set of operations. First, compute the transformed point set $F_p(\mathcal{P})$. Then, compute the construction with the Euclidean norm on $F_p(\mathcal{P})$ and transform the result back through F_p^{-1}. The triangulation $\mathrm{Del}_p(\mathcal{P})$ is thus simply the image through the stretching transformation F_p^{-1} of the Euclidean Delaunay triangulation $\mathrm{Del}(F_p(\mathcal{P}))$ where $F_p(\mathcal{P}) = \{F_p p_i, p_i \in \mathcal{P}\}$.

As explained in the Section 2.2 of Chapter 2, a sphere in metric space is an ellipsoid in the Euclidean space. Each simplex of a uniformly anisotropic Delaunay triangulation $\mathrm{Del}_p(\mathcal{P})$ thus possesses an empty circumscribing ellipsoid, the inverse-transformed Delaunay ball from metric to Euclidean space (see Figure 4.1).

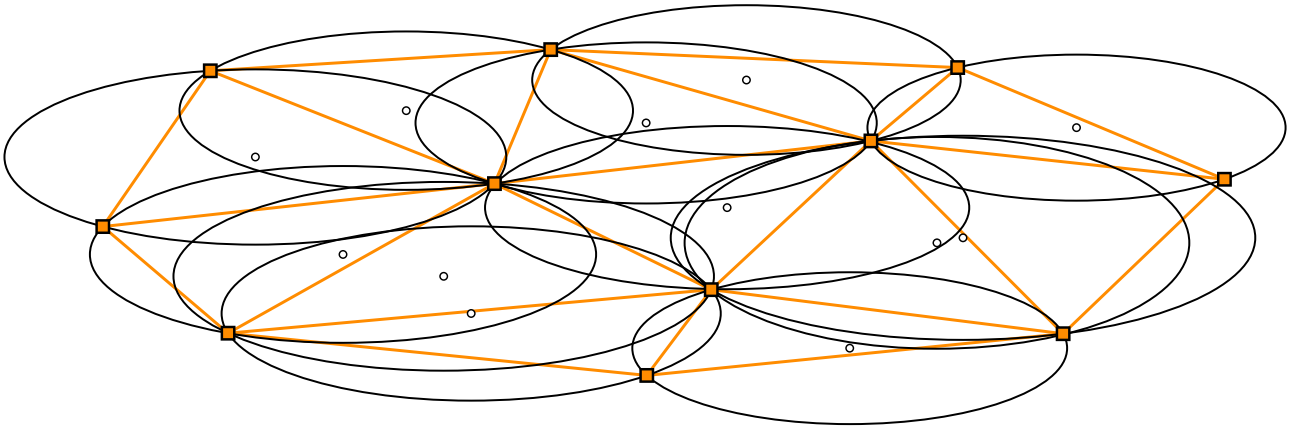

Figure 4.1: An anisotropic uniform Delaunay triangulation (orange) and the corresponding stretched Delaunay balls and circumcenters (black circles).

The central idea of the framework of locally uniform anisotropic meshes is to approximate at each vertex p a given arbitrary metric field g by the uniform metric defined by extending G_p over the domain (that is the metric field $g_0 : q \in \Omega \mapsto G_p$). Similarly to the way affine functions are locally good approximations of a generic continuous function, the approximation of an arbitrary metric field by a uniform metric will be accurate as long we stay in a small neighborhood. At each vertex, a Delaunay triangulation that conforms to the uniform metric field of that vertex is constructed. These independent triangulations can – under some density conditions – be combined to obtain a final triangulation of the domain.

4.1.1 Stars and inconsistencies

The *star* of a vertex p in a simplicial complex $\mathcal{K}$, denoted by S_p, is defined as the subcomplex of $\mathcal{K}$ formed by the set of simplices that are incident to p. The idea of considering independent stars at the

vertices of a point set was first conceived by Shewchuk [128] to handle moving vertices in finite element meshes. This structure was also employed by Schoen [126], who introduced anisotropic stars whose connectivity is obtained by building an isotropic Delaunay mesh of a transformed point set. The star S_p of $p \in \mathcal{P}$ is in that case extracted from the complex $\mathrm{Del}_p(\mathcal{P})$. This construction was described in the previous section and forms the core of the locally uniform anisotropic mesh framework. The collection of all the stars is called the (anisotropic) *star set* of $\mathcal{P}$ and is noted $\mathcal{S}(\mathcal{P})$.

As the connectivity of each star is set according to the metric G_p at the center of the star, a given n-simplex has $n+1$ different Delaunay balls, one with respect to the metric of each vertex of the simplex. Consequently, there are in general *inconsistencies* among the stars of the sites: a simplex τ, appearing in the stars of some of its vertices, may not appear in the stars of all of them (Figure 4.2). If a simplex is involved in such configuration, it is said to be *inconsistent*. Stars containing such simplices are *inconsistent stars* and a star set with at least one inconsistent star is also said to be inconsistent. Oppositely, a star whose simplices are consistent is said to be *consistent* and a star set is *consistent* if all of its stars are consistent.

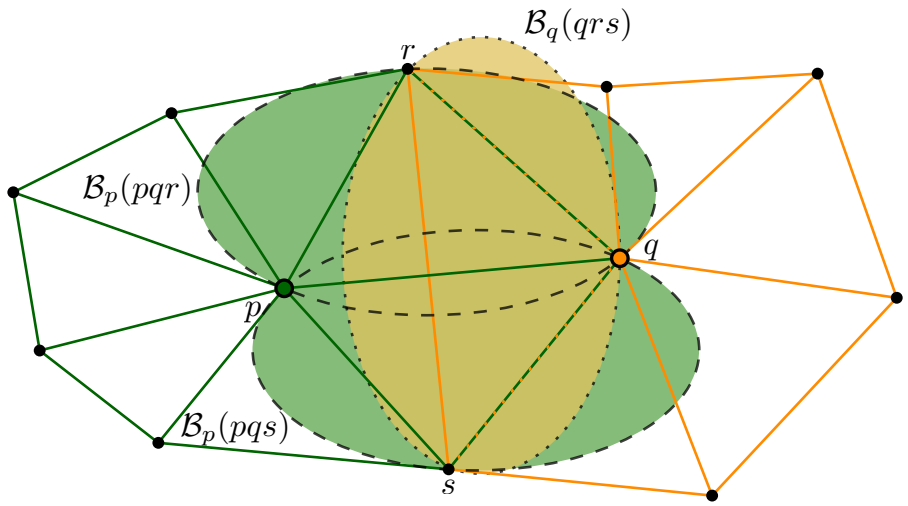

Figure 4.2: Two stars S_p and S_q forming an inconsistent configuration.

The main idea of the algorithm is to refine the set of sites $\mathcal{P}$ while maintaining the set of stars $\mathcal{S}(\mathcal{P})$ until each star S_p in $\mathcal{S}(\mathcal{P})$ is composed of simplices that are well shaped and well sized in the metric G_p, and until there are no more inconsistencies among the stars. Once a consistent star set is achieved (which is proven to happen), all the stars can be stitched into a single triangulation: a locally uniform anisotropic mesh that conforms to the specified metric field and offers guarantees on the quality of the simplices.

4.1.2 Slivers and quasi-cosphericities

To be able to merge a star set into a triangulation, the stars must have compatible connectivity; in other words, the star set must be consistent.

It can be shown (see Lemma 4.1.2) that once the point set $\mathcal{P}$ is dense enough such that the distortion between adjacent points is bounded, inconsistencies are only created by *quasi-cosphericities*, a degenerate geometrical configuration where $n+2$ points all lie almost on a $n+1$-sphere. Some of these quasi-cosphericities are related to slivers, well-known problematic elements of simplicial meshes [94, 127, 133]. Both notions are now recalled and the link between inconsistencies, quasi-cosphericities and slivers is explained.

Sliver The definition of a sliver with respect to a metric is derived from the isotropic definition of Cheng et al. [49]. For a k-simplex τ, the circumscribing G_p-ball of τ is given by $B_p(c_p(\tau), r_p(\tau))$ and the shortest edge G_p-length is given by $e_p(\tau)$. The *radius-edge ratio* is a classical indicator of the quality of a simplex and is defined as $\rho_p(\tau) = r_p(\tau)/e_p(\tau)$. The *sliverity ratio* $\sigma_p(\tau)$ is given by the ratio $\left(\mathrm{Vol}_p(\tau)/e_p^k(\tau)\right)^{\frac{1}{k}}$, where $\mathrm{Vol}_p(\tau)$ is the G_p-volume of τ. Two (positive) constant parameters, ρ_0 and σ_0, control how degenerate a simplex can be before it is called a sliver. The simplex τ is said to be

- *well-shaped* for G_p, if $\rho_p(\tau) \leq \rho_0$ and $\sigma_p(\tau) \geq \sigma_0$;
- a *sliver* for G_p, if $\rho_p(\tau) \leq \rho_0$ and $\sigma_p(\tau) < \sigma_0$;
- a k-sliver for G_p, if it is a sliver but all its $(k-1)$-faces are well-shaped.

Slivers are undesirable simplices as they can compromise the convergence and quality of numerical simulations.

Quasi-cospherical configurations Let ψ_0 be a bound on the distortion and G a metric. A subset $\mathcal{P}'$ of $n+2$ points is said to be a (ψ_0, G)-*cosphericity* if there exists two metrics G' and G'' such that

- $\psi(G, G') \leq \psi_0$, $\psi(G, G'') \leq \psi_0$ and $\psi(G', G'') \leq \psi_0$,
- the triangulations $\mathrm{Del}_{G'}(\mathcal{P}')$ and $\mathrm{Del}_{G''}(\mathcal{P}')$ are different.

A quasi-cosphericity involves multiple simplices, some of whom might be slivers. When the values of G and ψ_0 are obvious or irrelevant, the term *quasi-cosphericity* is used. Figure 4.3 shows a quasi-cosphericity involving 4 vertices.

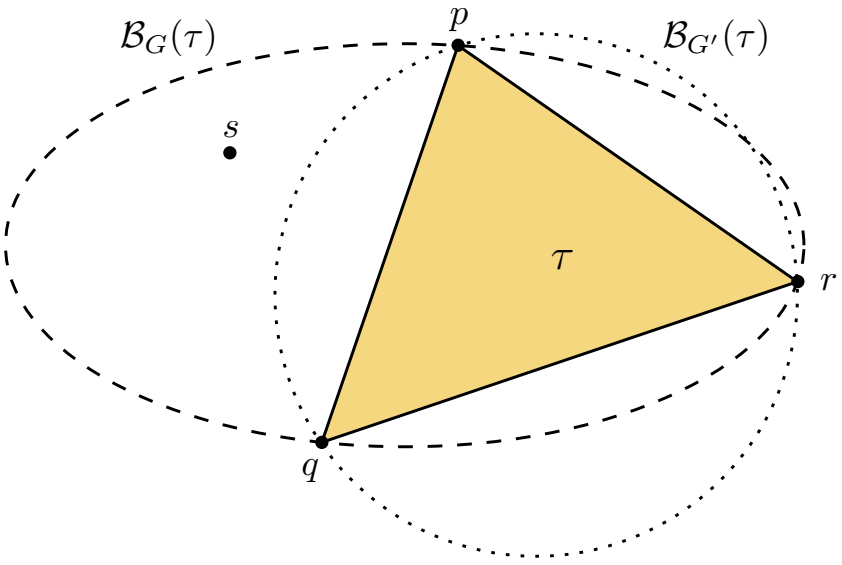

Figure 4.3: A quasi-cosphericity involving the simplex τ and the point s: s belongs in the $\mathcal{B}_G(\tau)$, but not in $\mathcal{B}_{G'}(\tau)$.

Remark 4.1.1 *In the definition of a cosphericity, all points play a symmetric role.*

The following lemma relates quasi-cospherical configurations and inconsistencies.

Lemma 4.1.2 *Let τ be an inconsistent simplex of star S_p. If $\psi(\tau) < \psi_0$, then there exists a site $q \in \mathcal{P}$ such that the configuration (τ, q) is (ψ_0, G_p)-cospherical.*

Proof. This result is a direct consequence from the definition of an inconsistency: an inconsistency is a geometrical configuration in which a simplex τ exists in a star S_p, but not in another star S_q (with $p, q \in \tau$). In other words, there exists a point r such that $r \in \mathcal{B}_q(\tau)$ but $r \notin \mathcal{B}_p(\tau)$, which is the definition of a quasi-cosphericity. □

The intent is thus to refine the star set while avoiding the apparition and presence of quasi-cosphericities in the star set: by contrapositive, the absence of quasi-cosphericities gives us the absence of inconsistencies when the mesh is dense. However, quasi-cosphericities that contain slivery simplices cannot be avoided (see the proof of the Picking Lemma in [31]); slivery simplices must therefore be eliminated first.

4.2 Refinement algorithm

The simplest idea to refine a simplex τ in a star S_p is to insert a new site at the center $c_p(\tau)$ of the Delaunay G_p-ball of τ. This technique is very common in Delaunay refinement algorithm as the Delaunay ball of the simplex is by construction not empty after the insertion of its center and thus the simplex cannot appear in the new Delaunay triangulation.

This simple strategy may unfortunately lead to cascading occurrences of inconsistencies, for the same reason that the refinement of Delaunay meshes cannot remove slivers. An alternative strategy is devised, inspired by the work of Li and Teng [95, 96] to avoid slivers in isotropic meshes. The adaptation of Li and Teng's techniques to the present algorithm and the development of the refinement algorithm are described in detail in the following.

4.2.1 The `Pick_valid` procedure

The central idea in the work of Li and Teng is to relax the choice of the refinement point of a simplex: the refinement point is no longer the center of the Delaunay ball, but is instead carefully chosen from a small *picking region* around this center. In the framework of locally uniform anisotropic meshes, the wiggle room in the choice of the refinement point allows to find refinement point who destroy inconsistent simplices without creating new quasi-cosphericities.

The region in which refinement points are picked is called the *G_p-picking region* of a simplex τ in the star S_p is defined as the G_p-ball $\mathcal{B}_p(c_p(\tau), \delta r_p(\tau))$ and is denoted by $\mathrm{P}_p(\tau)$. The constant parameter $\delta < 1$ is the *picking ratio*, which controls the size of this small region. A refinement point picked in the picking region is called a *candidate (point)*. These definitions are illustrated in Figure 4.5.

By construction, a candidate point p in $\mathrm{P}_q(\tau)$ is included in $B_q(\tau)$ since $\delta < 1$. The simplex τ is thus sure to be removed from S_q when p is inserted in $\mathrm{Del}_q(\mathcal{P})$ since τ is not a Delaunay simplex anymore. However, the insertion of p in the star set should not result in the creation of new slivers or inconsistencies. It is, in fact, not always possible to completely avoid the formation of new inconsistencies when choosing a refinement point in the picking region $\mathrm{P}_p(\tau)$ of a simplex τ of S_p. A refinement point $p \in \mathrm{P}_q(\tau)$, is said to be *valid* if the insertion of p in the star set does not trigger the appearance of *small* inconsistent facets, where a simplex τ' in S_q is considered small if the G_q-radius $r_q(\tau')$ of its Delaunay ball is less than $\beta r_q(\tau)$, for a given parameter $\beta \geq 1$.

The `Pick_valid` procedure is detailed in Algorithm 1.

Algorithm 1 `Pick_valid`(τ, G_q)

Step (1) Pick randomly a point p in the picking region $\mathrm{P}_q(\tau)$

Step (2) *Avoid small slivers*
if the insertion of p in some star S_r, creates a k-simplex τ', with $k \leq d$ such that τ' is a k-sliver ($\rho_r(\tau') \leq \rho_0$ and $\sigma_r(\tau') \leq \sigma_0^d$) with G_r-circumradius less than βr, **then**
discard p and go back to step 1.

Step (3) *Avoid small quasi-cosphericities*
if the point p forms an inconsistent d-simplex with distortion less than γ_0 in some star S_r, with a G_r-circumradius smaller than $\beta r_p(\tau)$, **then**
Discard p and go back to step 1.

Step (4) Return p.

As shown in the next sections, it can be proven that once the distortion of simplices is low enough, valid points exist in the picking region.

Forbidden zones

The `Pick_valid` algorithm tries to avoid the creation of small slivers and small quasi-cosphericities during the insertion of the refinement point. These bad geometrical configurations are created by combinations of vertices neighboring the refinement point. In $\mathbb{R}^n$, a set of $n+1$ vertices $\mathcal{S}$ creates regions of the domain where the refinement point should not lie in order to not create small slivers or small quasi-cosphericities. This region is called a *forbidden region*. By definition, the forbidden region of a simplex σ is the union of points in Ω that fall within at least one Delaunay ball of σ but do not fall within all the $n+1$ Delaunay balls. Due to the multitude of neighbors around a vertex, many forbidden regions usually exist.

To simplify theoretical purposes, we can encompass a forbidden region within two G-spheres (with G an arbitrary metric), as done in Figure 4.4. These two spheres create a *forbidden G-ring* of σ, which we denote $\mathcal{F}_G(\sigma)$.

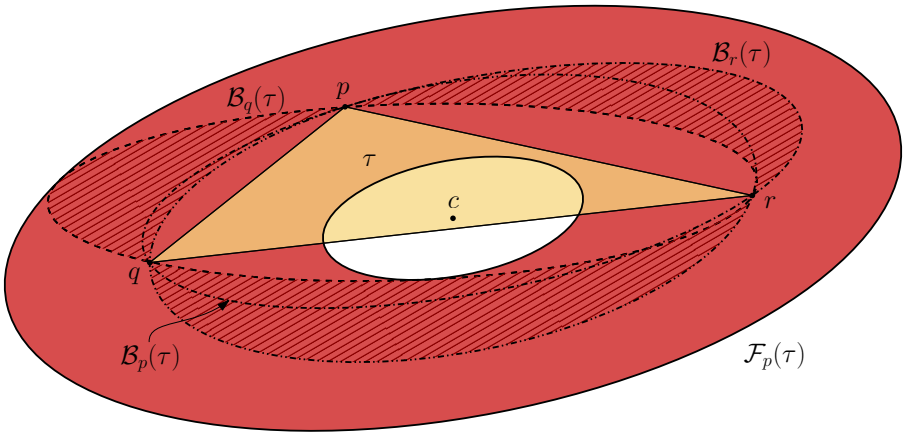

Figure 4.4: The forbidden region (dashed) and forbidden ring (colored red) of a simplex, illustrated in Euclidean space.

Remark 4.2.1 *The notion of forbidden ring depends on the choice of a metric, while the notion of forbidden region does not.*

When picking a point p to refine a simplex σ in a star S_q, if p does not belong to the G_q-ring of σ, then it does not create a quasi-cosphericity with the vertices of σ. The intersection of the forbidden G_q-ring and the picking region $\mathrm{P}_q(\sigma)$ of a simplex is called a *forbidden zone* of the picking region. There is by definition no valid point when forbidden zones cover entirely the picking region.

Figure 4.5 illustrates all the notions used in the `Pick_valid` algorithm.

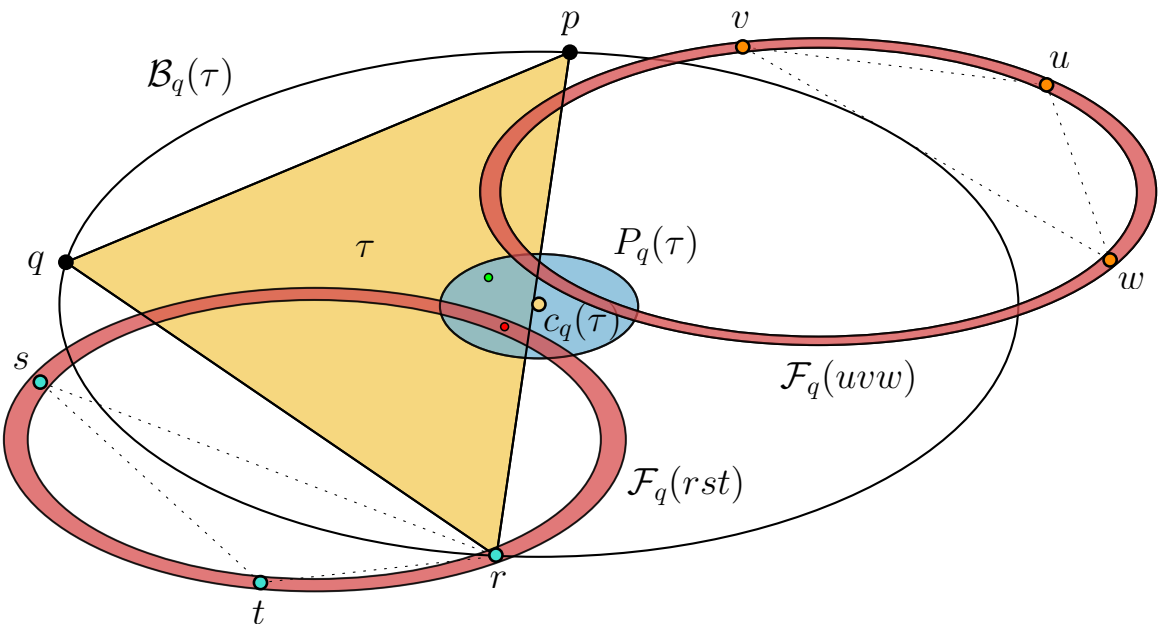

Figure 4.5: A simplex τ, its Delaunay ball $\mathcal{B}_q(\tau)$ (in the metric G_q) and the picking region $\mathrm{P}(\tau)$, in blue. Two forbidden rings (in red) are created by neighboring vertices, one created by s and t along with a vertex of τ, r; and one created by u, v, w. One possible valid refinement point in $\mathrm{P}_q(\tau)$ is drawn in green and one possible invalid point in red.

Thickness of a forbidden zone Simply applying the definition of a quasi-cosphericity, the forbidden ring associated to a simplex τ is a region of space where points fall in at least one but not al of the Delaunay balls of τ. Therefore, the area of a forbidden ring is proportional to the distortion between the metrics of the vertices of the simplex: the larger the distortion, the larger the difference between the Delaunay balls and the wider the forbidden ring. The `Pick_valid` procedure is thus unlikely to find a valid point when the distortions between neighboring vertices are large.

To improve the odds of finding a solution, the refinement algorithm ensures that the distortion is low between neighboring vertices before the `Pick_valid` procedure is called. This is achieved by imposing a maximum value on the distortion allowed within a simplex, where the distortion of a simplex τ – denoted by $\psi(\tau)$ – is the maximum distortion between any two vertices of τ. This maximum distortion is controlled a constant parameter of the algorithm, denoted by ψ_0.

A proper choice of the algorithm parameters ψ_0, δ, and β is essential to ensure the existence of valid refinement points in the picking regions (see Section 4.2.4).

4.2.2 Criteria

The refinement algorithm aims to produce a set of stars whose simplices are Delaunay, satisfy quality constraints and are consistent. We detail below the criteria that are used to drive the refinement process. These criteria can be split in two categories depending on their role.

Size and quality

As for classical isotropic refinement algorithms, the anisotropic refinement algorithm includes size and quality criteria.

- The size of simplices is controlled by comparing the circumradius of simplices with a user-defined sizing field sf. A simplex $\tau \in \mathrm{S}_p$ is *oversized* if the G_p-circumradius $r_p(\tau)$ of its Delaunay G_p-ball is greater than $\mathrm{sf}(c_p(\tau))$.

  Remark 4.2.2 *The metric field already incorporates a sizing field, rendering this parameter redundant. For this reason, we (almost) always consider* $\mathrm{sf} = 1$ *and the sizing constraint is modified directly within the metric field (see Section 2.13 in Chapter 2 for further details).*

- The quality of a simplex τ with respect to metric G_p of a vertex $p \in \tau$ is estimated through its radius-edge ratio $\rho_p(\tau)$, which measures the ratio between the G_p-circumradius and the G_p-length of the shortest edge of τ. A simplex $\tau \in \mathrm{S}_p$ is *badly shaped* if its radius-edge ratio $\rho_p(\tau)$ is greater than a constant parameter ρ_0.

Consistency

The purpose of the criteria described below is to help the resolution of inconsistencies.

- As we have explained above, inconsistencies are difficult to solve when the distortion is large within simplices. A simplex $\tau \in \mathrm{S}_p$ is *distorted* if its distortion $\psi(\tau)$ is greater than a constant parameter ψ_0.

- Once the distortion is low in all simplices, only quasi-cosphericities are responsible for the presence of inconsistencies in the star set. Quasi-cosphericities that involve slivers are difficult to deal with and thus removing slivers ensure the removal of this type of quasi-cosphericities and, by extension, of inconsistencies that result from these low-quality configurations (see Section 4.1.2). A simplex $\tau \in \mathrm{S}_p$ is *slivery* if its sliverity ratio $\sigma_p(\tau)$ is greater than σ_0.

Oversized, badly shaped, distorted, slivery and inconsistent simplices are *bad* simplices that must be destroyed.

4.2.3 Algorithm

Starting from a small set of initial vertices, points are iteratively inserted until all simplices satisfy a given set of criteria. While there exist bad simplices in the star set, the algorithm selects one such simplex τ and inserts a new point, called the *refinement point* or *Steiner point* of τ. Inserting a new point in the star set involves maintaining the connectivity of all the stars in the star set $\mathcal{S}(\mathcal{P})$.

For a distorted or oversized simplex τ in some star S_p, the refinement point is simply chosen to be the G_p-circumcenter $c_p(\tau)$ of the simplex. However, to guarantee the termination of the refinement

process, the algorithm must carefully choose the refinement point of a slivery or inconsistent simplex in order to avoid the cascading appearance of inconsistent configurations.

The algorithm relies on the `Insert` procedure (Algorithm 2) to insert a new point in the data structures, and on the `Pick_valid` procedure (see Section 4.2.1) to select the location of the new site when the refinement point is not the circumcenter.

Algorithm 2 `Insert`(p)

Create the new star S_p
Insert p in any star $\mathrm{S}_q \in \mathcal{S}(\mathcal{P})$ where p will appear as a vertex

A greedy approach is employed: the rules of Algorithm 3 are applied with a priority order, meaning that Rule (i) is applied only if no Rule (j), with $j < i$, can be applied.

Algorithm 3 Refinement algorithm

Rule (1) **Distortion:**
If $\exists\, p \in \mathcal{P}$ and $\tau \in \mathrm{S}_p$ such that $\psi(\tau) \geq \psi_0$,
then `Insert`$(c_p(\tau))$;

Rule (2) **Size:**
If $\exists\, p \in \mathcal{P}$ and $\tau \in \mathrm{S}_p$ such that $r_p(\tau) \geq \mathrm{sf}(c_p(\tau))$,
then `Insert`$(c_p(\tau))$;

Rule (3) **Shape:**
If $\exists\, p \in \mathcal{P}$ and $\tau \in \mathrm{S}_p$ such that $\rho_v(\tau) > \rho_0$,
then `Insert`(τ, G_p);

Rule (4) **Sliver elimination:**
If $\exists\, p \in \mathcal{P}$ and $\tau \in \mathrm{S}_p$ such that τ is a sliver (*i.e.* $\rho_v(\tau) \leq \rho_0$, $\sigma_p(\tau) < \sigma_0$),
then `Insert`(`Pick_valid`(τ, G_p));

Rule (5) **Consistency:**
If $\exists\, p \in \mathcal{P}$ and $\tau \in \mathrm{S}_p$ such that τ is inconsistent,
then `Insert`(`Pick_valid`(τ, G_p));

The algorithm ends when no rule applies anymore, which is proven to happen (see Section 4.2.4). The star S_p of any vertex $p \in \mathcal{P}$ is Delaunay for the metric G_p of its central vertex and since the criteria are satisfied, all the simplices of S_p have guaranteed size and shape quality with respect to G_p. Lastly, the stars are consistent and can be stitched together to form a triangulation $\mathcal{D}(\mathcal{P})$, which is thus a locally uniform anisotropic mesh whose simplices have guaranteed quality in the local metric.

4.2.4 Termination

The termination of the algorithm is proved by showing that the radius of insertion, the minimum distance between a point being inserted in the star set and the existing vertices of the star set, is lower bounded and observing that since the domain has finite volume, the number of vertices must be bounded. Since we insert a vertex at each iteration, the refinement process must therefore finish.

An important step to obtain the termination of the algorithm is to show that the `Pick_valid` cannot fail indefinitely as the mesh gets denser. Recall that the picking region of a simplex, a scaled

down version of the Delaunay ball of the simplex, is intersected by forbidden rings, which are composed of candidates points that would create (small) quasi-cosphericities with nearby vertices and are thus invalid (see Section 4.2.1). It can be proven that since refinement points are taken near the circumcenters of simplices, there is only a finite and relatively constant number of forbidden rings that can intersect a given picking region. Additionally, the thickness of a forbidden ring is linked to the distortion of the simplex and vanishes as the distortion goes to 1 (see Section 4.2.1). By another volume argument, the procedure must eventually finds valid points.

Details can be found in the thorough theoretical study of the algorithm by Boissonnat et al. [31]. Note that the final step is missing in their analysis: a consistent star set does form an abstract triangulation, but it must be shown that it is a good triangulation of the input domain. This result can nevertheless be obtained using Boissonnat et al. [15, Theorem 7.13].

4.3 Mesh generation of domains

We have so far recalled a refinement algorithm (Algorithm 3) that constructs a triangulation satisfying a set of criteria from an initial set of points. However, practical mesh generation must be able to handle bounded domains and to capture the boundary precisely. In the following, we assume that the domain Ω is embedded in $\mathbb{R}^3$ and bounded by a smooth 2-manifold (a surface) $\mathcal{M} = \partial\Omega$. This is only done to simplify the presentation and the concepts and algorithms detailed below could be applied to higher dimensions, and even to the case of manifolds of arbitrary co-dimension.

In the context of isotropic mesh generation, the use of restricted Delaunay triangulations combined with a criteria-based meshing algorithm has been proved to create meshes that are faithful approximations of the boundary [53, 27]. Additionally, the definition of restriction to a domain allows to easily mesh implicit surfaces, polyhedral surfaces, and any other type of surfaces, provided that we are able to know if a dual entity intersects the domain (or the surface) and to compute the intersection point. This powerful approach is thus employed and adapted here to produce good anisotropic triangulations of domains.

We now describe how to create an anisotropic mesh of Ω, based on the refinement algorithm previously described.

4.3.1 Restricted and surface stars

As we construct at each point a Delaunay triangulation, it is natural to consider the restriction of each of these Delaunay triangulations, and specifically the restriction of stars. The notion of restricted Delaunay complex is thus smoothly applied on the structure of the star set.

To each star S_p is associated the *volume star* S_p^v, formed by the cells in S_p whose dual Voronoi vertex belongs to Ω. The set of volume stars is called the *volume star set* and is denoted by $\mathcal{S}^v(\mathcal{P})$. Similarly, we associate to each star S_p the *surface star* S_p^s, formed by the facets in S_p whose dual Voronoi edge intersects the boundary $\mathcal{M}$. Each facet σ of a surface star S_p^s has a G_p-surface Delaunay ball, *i.e.* an G_p-circumscribed ball centered on the surface and including no point of $\mathcal{P}$. The surface G_p-circumball of σ is denoted by $B_p(\sigma)$, its radius $r_p(\sigma)$ and its center $c_p(\sigma)$, which is the intersection of the dual of σ with the surface. The set of surface stars is called the surface star set and is denoted by $\mathcal{S}^s(\mathcal{P})$.

Similarly to the first refinement algorithm that we recalled, there are in general *inconsistencies* among restricted stars. Indeed, A simplex τ that appears in the volume stars of some of its vertices may not appear in the volume stars of all of them. Likewise, a simplex σ that appears in the surface

stars of some of its vertices may not appear in the surface stars of all of them. The sets of stars $\mathcal{S}^v(\mathcal{P})$ and $\mathcal{S}^s(\mathcal{P})$ thus do not necessarily form a triangulation of the domain or of the boundary.

Remark 4.3.1 *It should be emphasized that to obtain a triangulation, there must be consistency of both the volume and the surface star sets. While it is relatively easy to obtain a consistent surface star set without having a consistent volume star set, it is also possible for a tetrahedron to be consistent and to have a restricted facet that is not consistent. This results from the fact that whether a restricted facet is consistent or not does not depend purely on combinatorics but also whether it is restricted in the stars of all its vertices.*

As in Algorithm 3, it is necessary to cleverly select refinement points to obtain the consistency of both star sets. The picking region of a volume star is defined as in Section 4.2.1. The *G_p-picking region* of a facet σ in S_p^s, denoted by $\mathrm{P}_p(\sigma)$, is slightly more complex and is the intersection of the G_p-ball $\mathcal{B}_p(c_p(\sigma), \delta r_p(\sigma))$, with $0 < \delta < 1$, and of the surface $\mathcal{M}$ (see Figure 4.6).

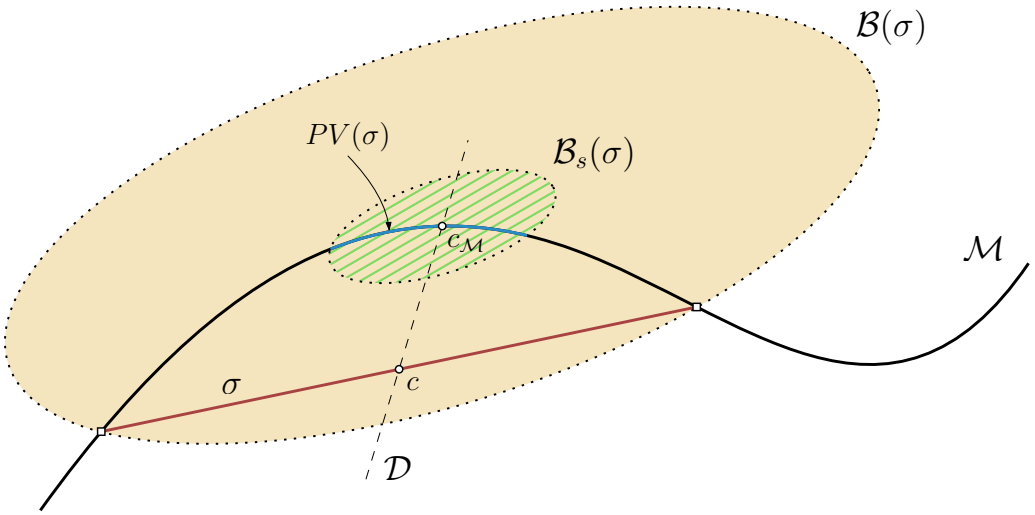

Figure 4.6: The picking region of a surface facet σ in the case $n = 2$ and $m = 1$. $\mathcal{D}$ is the dual of σ, $\mathcal{B}$ its surface Delaunay ball, and $c_{\mathcal{M}}$ the center of $\mathcal{B}$. The scaled down ball $\mathcal{B}_s$ has radius $\delta r_{\mathcal{B}}$ where $r_{\mathcal{B}}$ is the radius of $\mathcal{B}$.

4.3.2 Algorithm

The refinement process must now handle two different star sets, $\mathcal{S}^v(\mathcal{P})$ and $\mathcal{S}^s(\mathcal{P})$. The rules and criteria for volume stars are identical to those defined for Algorithm 3: distortion, size, shape, sliverity, and inconsistency. Some of these rules are identical for the surface star set – size, shape, consistency – but two new rules, specific to the boundary, are also introduced:

- An approximation rule is introduced to control the maximum distance between facets and the surface. The *approximation error* of a facet σ, denoted by $h_p(\sigma)$ is computed as the Euclidean distance from the barycenter of σ to the surface. As the approximation and size criteria are related, the approximation is incorporated within the sizing criterion.
- The second rule handles topological defects, which are defined as facets σ for whom not all the vertices are on the surface $\mathcal{M}$. This condition is an important indicator of the conformity of the mesh to the topology of the input domain (see Rineau [120]).

Remark 4.3.2 *The sliverity rule does not make sense for facets in the case of a surface embedded in* $\mathbb{R}^3$*, but would be required in the case of a manifold of higher dimension.*

The refinement algorithm for a domain with smooth boundary is presented in Algorithm 4. Rules are here again applied with a priority order: Rule (i) is applied only if no Rule (j), with $j < i$, can be applied.

Algorithm 4 Refinement algorithm for a domain with a smooth boundary

Rule (1) **Facet distortion:**
If $\exists p \in \mathcal{P}$ and $\sigma \in \mathrm{S}_p^s$ such that $\psi(\sigma) \geq \psi_0$,
then `Insert`$(c_p(\sigma))$;

Rule (2) **Facet size:**
If $\exists p \in \mathcal{P}$ and $\sigma \in \mathrm{S}_p^s$ such that $r_p(\sigma) \geq \mathrm{sf}(c_p(\sigma))$ or $h(\sigma) \geq h_0$,
then `Insert`$(c_p(\sigma))$;

Rule (3) **Facet topology:**
If $\exists p \in \mathcal{P}$ and $\sigma \in \mathrm{S}_p^s$ such that σ has a vertex q not on $\mathcal{M}$,
then `Insert`$(c_p(\sigma))$;

Rule (4) **Facet shape:**
If $\exists p \in \mathcal{P}$ and $\sigma \in \mathrm{S}_p^s$ such that $\rho_p(\sigma) > \rho_0$,
then `Insert`$(c_p(\sigma))$;

Rule (5) **Tetrahedron distortion:**
If $\exists p \in \mathcal{P}$ and $\tau \in \mathrm{S}_p^v$ such that $\psi(\tau) \geq \psi_0$,
then `Insert`$(c_p(\tau))$;

Rule (6) **Tetrahedron size:**
If $\exists p \in \mathcal{P}$ and $\tau \in \mathrm{S}_p^v$ such that $r_p(\tau) \geq \mathrm{sf}(c_p(\tau))$,
then `Insert_or_snap`(τ, G_p));

Rule (7) **Tetrahedron shape:**
If $\exists p \in \mathcal{P}$ and $\tau \in \mathrm{S}_p^v$ such that $\rho_p(\tau) > \rho_0$,
then `Insert_or_snap`(τ, G_p);

Rule (8) **Tetrahedron sliver removal:**
If $\exists p \in \mathcal{P}$ and $\tau \in \mathrm{S}_p^v$ such that $\rho_p(\tau) \leq \rho_0$ and $\sigma_p(\tau) < \sigma_0$),
then `Insert_or_snap_valid`(τ, G_p);

Rule (9) **Facet Consistency:**
If $\exists p \in \mathcal{P}$ and $\sigma \in \mathrm{S}_p^s$ such that σ is inconsistent,
then `Insert`(`Pick_valid`(σ, G_p));

Rule (10) **Tetrahedron consistency:**
If $\exists p \in \mathcal{P}$ and $\tau \in \mathrm{S}_p^v$ such that τ is inconsistent,
then `Insert_or_snap_valid`(τ, G_p);

The `Insert_or_snap` and `Insert_or_snap_valid` procedures are explained and detailed below in Section 4.3.3.

Note that the rules – whether surface or volume rules – that rely on the `Pick_valid` procedure to select the refinement point are assigned the lowest priority, in hope that a slivery or inconsistent simplex is destroyed by a criterion that does not require the use of the costly `Pick_valid` procedure.

Pure mesh generation Algorithm 4 describes the process that refines both the inner region and the boundary of the domain. It is however possible to ignore the refinement of one of the two types of stars: ignoring volume star rules yields a consistent surface mesh of the domain, and ignoring surface rules yields a triangulation of the domain (but whose boundary is usually roughly approximated). When one of the star set is ignored, we use the keyword *pure*: either a pure surface, or a pure volume mesh is constructed.

4.3.3 Encroachment

While refining a volume star, a chosen Steiner point p might fall in a surface Delaunay G_p-ball of a facet $\sigma \in \mathrm{S}_p$. If it is the case, q is said to *encroach* σ. If a refinement point is to be inserted to destroy a cell and and it encroaches a surface fact, then the facet is refined instead. The insertion of points in volume stars is thus done through two procedures that are modified version of the `Insert` and `Pick_valid` procedures to incorporate the possibility of encroachment: `Insert_or_snap` (Algorithm 5) and `Insert_or_snap_valid` (Algorithm 6).

Algorithm 5 `Insert_or_snap`(τ, G_q)

Let $c = G_q$-circumcenter of τ
if c encroaches some facet σ in some restricted surface star S_r^s **then**
    `Insert`$(c_r(\sigma))$
else
    `Insert`(c)

Algorithm 6 `Insert_or_snap_valid` (τ, G_q)

Let $c =$ `Pick_valid` (τ, G_q)
if c encroaches some facet σ in some restricted surface star S_r^s **then**
    `Insert`(`Pick_valid` (σ, G_r))
else
    `Insert`(c)

4.3.4 Features preservation

In practice, three-dimensional domains are rarely bounded by a smooth surface, but rather possess a piecewise-smooth boundary. A mesh of the domain is then expected to include a faithful representation of the 1-dimensional features of the boundary. A natural idea would be to create a third type of star set, for the edges, with its own set of rules, along with a notion of encroachment between facets and edges, as was done by Rineau [120]. However, this technique fails in the isotropic setting when small angles exist between these features and this configuration can also occur in our context (with small angles computed in the metric).

To remedy this problem, the technique of protecting balls, inspired by Cheng et al. [46, 47], was used to handle 1-dimensional features robustly in the isotropic setting. The method could be extended to the framework of locally uniform anisotropic meshes by considering anisotropic balls, but this approach was not pursued in theory nor in practice due to unrelated shortcomings of the algorithm, see Sections 4.7 and 4.8.

4.3.5 Initial sampling

The algorithm requires a few initial points to start the refinement process. We generate a rough isotropic mesh and a sample of its vertices are extracted to form the initial set of sites of the anisotropic star set.

Poles When constructing a pure surface mesh, it was recommended to add a few initial points on the medial axis in addition to the initial points on the surface, called *poles*, for the sake of numerical robustness [28]. A good way to obtain those poles is to use the Voronoi vertices of the initial isotropic mesh; Figure 4.7 shows in yellow the poles corresponding to an isotropic triangulation of the "Dino" surface.

Figure 4.7: The "Dino" surface (leftmost). An isotropic mesh of the "Dino" surface (middle and rightmost) and its poles (yellow).

In practice, experiments show that poles are never required for robustness with our implementation and these degeneracies are thus a theoretical concern. Worse, the use of poles can even be harmful if they are close to the boundary of the domain as this will create a local over-refinement spot due to the shape criterion.

However, poles proved useful when both volume and surface rules are used, as they limit the size of the initial tetrahedra within the domain. Since the surface stars are first refined in Algorithm 4, this plays an important role in the speed of the algorithm: without poles, volume stars might possess an extremely large number of vertices once the size and shape surface rules are satisfied. By employing well-positioned poles within the domain, we can ensure that the number of vertices on the link stays low, thus greatly reducing the running time of the first few insertions, which would otherwise be large.

Remark 4.3.3 *This insertion of well-spaced interior points to ensure a good run time of the algorithm is not unlike the techniques proposed by Miller et al. [104] to produce so-called linear-size meshes.*

4.4 Implementation

The algorithm is straightforward and a naive implementation would be trivial: simply build all the complete Delaunay triangulations at all the vertices, and create the final mesh by extracting each star S_p from the triangulation $\mathrm{Del}_p(\mathcal{P})$. A new point q would be inserted in all the triangulations $\mathrm{Del}_p(\mathcal{P})$, regardless of the proximity between p and q, and the new star S_q is obtained by building $\mathrm{Del}_q(\mathcal{P} \cup q)$. Such an approach is clearly unreasonable in practice, both in terms of memory and time: the simple insertion of one point in the star set structure would carry an $O(n \log n)$ complexity. We therefore aim to implement the locally uniform anisotropic mesh theory efficiently and robustly.

Our code is based on the `C++` library CGAL [2], and specifically on the Triangulation packages that provide fast and robust implementations of the Delaunay, weighted Delaunay and constrained Delaunay triangulations. The framework of our implementation is inspired by that of the isotropic mesh generators of CGAL [80] and is designed to be as generic as possible and is thus heavily templated. We describe our current implementation in detail, from its general structure to the smaller tricks used to increase the speed of the algorithm.

The algorithm is implemented for planar domains, surfaces and volume domains with smooth boundaries.

4.4.1 General structure of the implementation

The core architecture of the implementation is similar to that of the isotropic mesh generators of CGAL, Mesh_2 and Mesh_3, and can be divided into three large sets of classes:

Data structures

A star is represented through a class called `Stretched_Delaunay_Triangulation` that derives from the CGAL class `Delaunay_Triangulation`. It provides the basic operations that a star requires such as the insertion and removal of a point, the computation of cavities and the combinatorial predicates used to answer queries such as "is the simplex σ found in this star?". The star set is, as its name implies, represented by a `vector` of stars and each star is assigned an index, its position in the star set. A `Star_set` class handles the insertion and removal of a point in the star set, the verification of consistency between stars, and various miscellaneous tasks such as the input and output functions.

Refinement

Refinement-related classes are largely independent from the classes that implement data structures. A class `Criteria` is in charge of providing the test functions associated with the different criteria, for example the computation of the circumradius and the comparison of the circumradius with the sizing parameter. Refinement queues are given their own specific classes, which contain the different priority queues for each criterion, and provide the insertion and removal of bad simplices from queues, as well as various query operations such as returning the worst element across all queues (which gives the next simplex to refine). More details are given on the efficient implementation of the refinement queue in Section 4.4.3.

Input parameters

The last set of classes concerns the two principal inputs: the domain and the metric field. The implementations of these inputs provide each one important function to the refinement algorithm, referred to *oracles*.

The class that implements the domains answers whether a point is inside, on the border or outside of the domain. This oracle is then used to compute whether a volume simplex is restricted, or to compute the intersection of the dual of a facet (either a ray or a segment) with the boundary of the domain, to test whether a facet is restricted.

The class that implements the metric fields computes the metric at any point of the domain.

Both classes are made as generic as possible by providing two base classes `Domain` and `Metric_field` that implement the basic operations. For example, the `Metric_field` can compute the stretching transformation of geometrical entities (points, simplices, boxes, etc.). Any domain or metric field is easily constructed by deriving from the respective base classes and only providing the specific oracle. All the metric fields and domains that are used in our experiments are presented in detail in Appendices A and B respectively.

We have so far described a number of classes that implement independent functionalities of the process. A skeleton is needed to link all these parts together; this is the topic of the next section.

4.4.2 Mesher levels

The process of inserting a point in the star set to destroy a simplex can be seen as a *cycle* of the refinement algorithm. The different rules of the algorithm (see Section 4.3.2) naturally split the set of cycles in *surface cycles* and *volume cycles*, depending on the kind of simplex that is refined during the cycle. Surface and volume cycles share a common core of tasks: retrieve the next bad simplex, compute the refinement point, insert the refinement point, maintain the star set $\mathrm{S}(\mathcal{P})$ and actualize the list of bad simplices.

These common tasks involve all the classes that were described in the previous section are involved: the simplex to destroy is selected from the refinement queues; the position of the refinement point is computed using the domain and its metric by the metric field; the insertion naturally is performed within the data structure classes; finally, gathering the new bad simplices is done with the help the Criteria classes and the insertion of bad simplices requires once again the refinement queue classes.

In the context of isotropic mesh generation, Rineau and Yvinec [119] introduced the notion of *Mesher level*, a class that creates the link between the different classes involved in a refinement algorithm and implements the different tasks shared by all types of cycle.

However, surface and volume cycles differ in the manner in which they perform some of these tasks; for example, the refinement point for a volume cycle is the center of a tetrahedron while the refinement point of a surface facet is the center of its surface Delaunay ball. Furthermore, tasks might need to be performed differently even within the same type of cycle, depending on the rule that is governing the insertion of the point: a tetrahedron that is a sliver calls upon the `Pick_valid` algorithm to determine the refinement point while an oversized tetrahedron is simply refined with its circumcenter.

It is thus necessary to create multiple mesher levels, each corresponding to a set of rules that perform the different tasks in the same manner. In the case of the rules of algorithm 4, there are thus four mesher levels:

- A facet level that handles the distortion, size and approximation, topology and shape of facets – Rules $(1-4)$.
- A cell level that handles Distortion, size, shape and sliverity of cells – Rules $(5-8)$.
- A facet consistency level that handles the consistency of facets – Rule (9).
- A cell consistency level that handles consistency of cells – Rule (10).

The complete refinement algorithms is obtained by stacking and tying together those mesher levels, according to the priority order of the rules: each mesher level can access the first mesher level with higher priority. If a mesher level M_1 is told to process an element (*i.e.* to perform the tasks of one

cycle) and a mesher level M_2 with higher priority still has simplices to refine, M_1 will not start a cycle and M_2 will process an element instead. Note that the mesher level M_2 will itself check if the next mesher level with higher priority has simplices to refine before starting the cycle. The refinement process is started by selecting the mesher level with lowest priority (usually a consistency mesher level) and telling it to process one element.

Communication from a higher priority mesher level to a lower priority mesher level is also sometimes needed, for example during the actualization of the refinement queues at all levels. To avoid problems in the initialization of the mesher levels due to circular dependency, the concept of *visitor patterns* – described in [3] – is used: to each mesher level is associated one visitor that points to the first mesher level with lower priority. Rather than a member of the mesher level, a visitor is passed by argument to the functions of that mesher level. Figure 4.8 shows the connectivity between mesher levels for Algorithm 4.

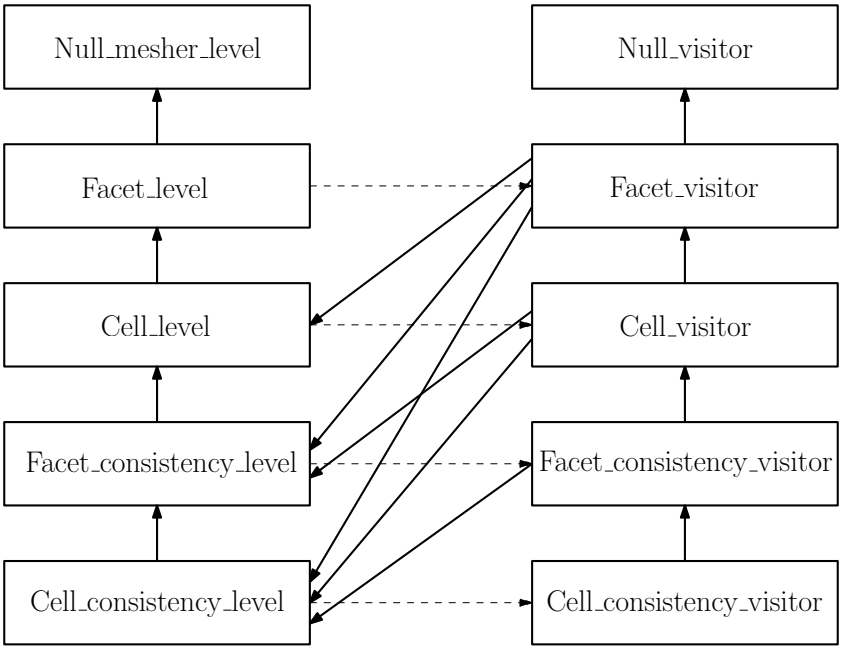

Figure 4.8: Pipeline of the hierarchical refining algorithm presented in Algorithm 4. "Null" mesher levels are empty mesher levels that are only needed for consistency. The dashed arrows signify that the functions of the mesher level are provided the corresponding visitor.

As mesher levels share a large amount of identical functions and a number of tasks with similar purpose but different execution, the usage of virtual functions impose itself naturally. The implementation of the mesher levels is based on the *Curiously Recurring Template Pattern* (CRTP), described by [54], to avoid the runtime cost of virtual functions.

The mesher level framework provides a powerful implementation of the algorithm as one can easily insert, modify, or change the priorities of a rule. For example, a pure surface mesher or a pure tetrahedral mesher can be obtained immediately by linking the mesher levels differently.

4.4.3 Refinement queues

The algorithm uses a greedy approach to refine simplices that do not satisfy the set of criteria. Each rule r is attributed a priority queue $\mathcal{Q}_r$ of simplices that are bad for this rule and that is sorted decreasingly, such that the simplex that is the worst for the criterion corresponding to r has the

highest priority in the queue. For example, oversized simplices (Rule (1)) are sorted by decreasing size of circumradius. Retrieving the next bad simplex to refine is done by finding the top of the refinement queues, that is the worst simplex in the highest priority queue amongst the non-empty queues. The bad simplex is removed from its queue once the insertion has been performed.

While the insertion of a point p that refines a bad simplex σ might destroy additional bad simplices $\{\sigma_j\}_j$, only σ is removed from the refinement queue at the end of a cycle. The top of the refinement queue might thus be a simplex that does not exist anymore. If that is the case, the top of the refinement queue is removed and we consider the new top of the refinement queue (and so forth until either the top is a valid simplex or the queue is empty). Maintenance functions to retrieve the next element to refine, insert elements or print the state of queues are implemented in a straightforward way.

The simplest implementation of the refinement queues is to use existing data structures and implement each rule using – for example – an `std::priority_queue` structure. However, the star set is a combination of many triangulations with each simplex living in multiple stars, creating redundancy in these queues. This issue is detailed in the next section, and an improvement to the naive implementation of the refinement queue is sought.

Redundancy

Due to the star-based structure of our algorithm, the same combinatorial simplex usually exists in multiple stars – and even in the star of all its vertices if it is consistent. On the other hand, a bad simplex σ will often be bad in more than one star:

- the distortion of a simplex is the same for all the metrics at its simplices. If the simplex is distorted, it will be immediately distorted for all the stars in which it appears. A similar reasoning can be applied to the topology and consistency criterion.
- if the distortion within a simplex is small, the evaluations of size, shape, approximation or sliverity will yield similar results in all the stars in which the simplex exists. If the simplex is bad for one star, it is likely to be bad for the others.

Furthermore, a bad simplex that is refined in one star (or destroyed by the refinement of another simplex) is probably also destroyed in the other stars of its vertices when the refinement point is inserted in the star set, but is only removed once from the refinement queues. Naively adding all the bad simplices to the corresponding priorities queues thus results in a lot of redundancy at different times.

This concern is not only theoretical: running the three dimensional example of a cube of side 3 endowed with a unidimensional shock metric field (see Section A.8 in Appendix A), we observe that after 3000 vertex insertions in the star set, the refinement queue is composed of 112437 different – maximal – simplices but the insertion of a bad simplex that already existed in the refinement queue has been performed 87064 times.

To prevent superfluous information in the refinement queues, we introduce a set that contains all the bad simplices with a uniqueness condition on the combinatorial information of simplices. We detail this approach and the necessary changes in the refinement queue implementation that go with it.

Data structures and complexity

We detail in this section the data structures and implementation of our improvement refinement queues.

Note first that a bad simplex σ in a star S_p is fully characterized by the following variables:

- S_p – the star in which it is considered,
- $\mathcal{C}_\sigma$ – the indices of its vertices in the star set,
- i_σ – the index of the rule for which it is bad,
- b_σ – its badness score for that rule.

Bad simplex are implemented with a class called `Bad_simplex`.

To preserve the uniqueness of a bad simplex in the refinement queues, a set of bad simplices Σ is built using an (unordered) set of `Bad_simplex`. A combinatorial simplex is thus allowed to appear only once. If a simplex is bad in multiple stars, the version that is kept in the refinement queue is the bad simplex that is bad for the rule with the highest priority. If the same simplex is bad for the same rule in multiple star – which often happens – the occurrence with the worst badness score is kept.

In practice, each priority queue $\mathcal{Q}_r$ (associated to the rule r) is a multiset of iterators *sit*, each pointing to an element σ in Σ, and is sorted by the values v_σ associated to the entry σ. For practical reasons, the position of a bad simplex σ in the queue $\mathcal{Q}_{r_\sigma}$, an iterator qit_σ of multiset, is also a member of the class `Bad_simplex`. The addition and removal of a bad simplex in the refinement queue are detailed in Algorithms 7 and 8 respectively.

Algorithm 7 Insertion of a bad simplex $\sigma(\mathrm{S}_p, \mathcal{C}_\sigma, i_\sigma, b_\sigma)$ in the refinement queue

if $\exists \sigma' \in \Sigma$ such that $\mathcal{C}_\sigma = \mathcal{C}_{\sigma'}$ **then**
  if $r_\sigma < r_{\sigma'}$ **or** $r_\sigma = r_{\sigma'}$ and $b_\sigma > v_{\sigma'}$ **then**
    Delete σ' from the refinement queue
    Insert σ in Σ
    Insert σ in $\mathcal{Q}_{i_\sigma}$
else
  insert σ in Σ

Algorithm 8 Deletion of a bad simplex $\sigma(\mathrm{S}_p, \mathcal{C}_\sigma, i_\sigma, b_\sigma)$ from the refinement queue

  Remove σ from Σ
  Remove σ from $\mathcal{Q}_{i_\sigma}$

Using standard structures from the Boost library, our refinement queue enjoys average constant complexity for the insertion and deletion of a bad simplex in the set Σ, and for the deletion of an entry from the refinement queue. The only operation that has non constant – or average constant – complexity is the insertion of an iterator sit_σ in the $\mathcal{Q}_{r_\sigma}$. For this operation, the complexity is logarithmic in the size of $\mathcal{Q}_{r_\sigma}$, which is thus also the complexity of the insertion of a bad simplex in the refinement queue.

In the following, we assume that our implementation uses this structure and will explicitly use the term *naive* when referring to the simpler implementation that ignores duplicates (and was described at the beginning of this section).

Maintaining the refinement queue

Once a point p has been inserted in the star set(s) $\mathcal{S}^v(\mathcal{P})$ and $\mathcal{S}^s(\mathcal{P})$, the refinement queue has to be updated. The bad simplex σ_0 that was killed by the insertion of p is removed as explained in algorithm 8. The last step of a cycle is to collect the new bad simplices and insert them in their respective refinement queues. Since it is not reasonable to test all the simplices of all the stars of the star set, we must gather a list (as small as possible) of $\mathcal{L}$ of potentially bad simplices. To build the

list $\mathcal{L}$, we gather the simplices of the new star and the simplices of the stars that were modified by the insertion of p. Unfortunately it is not sufficient to guarantee that we will find all the new bad simplices.

Tricky situations It might happen that a simplex that was consistent before the insertion of p has been destroyed in at least one – but not all – of the triangulations it appeared in. This simplex is now inconsistent but the stars in which it still appears might not have been modified by the insertion of p and thus has not been collected by $\mathcal{L}$. Note that this issue will appear even if using a naive implementation of the refinement queues.

Imposing the combinatorial uniqueness of elements greatly complicates the task of maintaining the refinement queues. Consider for example a bad simplex $\sigma(\mathrm{S}_\sigma, \mathcal{C}_\sigma, r_\sigma, v_\sigma)$ that is different from σ_0 (the simplex destroyed by the insertion of the new point p), but also has been destroyed by the insertion of p. The same (combinatorial) simplex might still exist and be bad in at least one of the stars of its vertices, giving another bad simplex $\sigma(\mathrm{S}_{\sigma'}, \mathcal{C}_{\sigma'}, r_{\sigma'}, v_{\sigma'})$ – with $\mathcal{C}_\sigma = \mathcal{C}_{\sigma'}$. Even though σ is not a valid entry in the queue anymore, it has not been removed from the refinement queue (only σ_0 has been). Therefore, σ' will not be inserted since it has lower priority than σ (since σ was in the queue, not σ'). This creates an obvious problem as σ will be discarded (since it doesn't exist anymore) when it is retrieved as the next simplex to refine, but σ' might still be an existing bad simplex that has never been marked for refinement. We must therefore preemptively remove the destroyed simplex σ from the refinement queue and add σ‘ to $\mathcal{L}$.

These two tricky situations are solved by complex – but relatively cheap – combinatorial checks in modified stars.

Finally, all the simplices of $\mathcal{L}$ are tested against the criteria of the algorithm, and inserted in the appropriate refinement queues if they are found to be bad.

Naive vs unique implementation of the refinement queues

Unfortunately, only tiny gains in speed were observed in practice as a result of our improved queue. This result is not completely unexpected as operations related to the refinement queue have in average constant or logarithmic complexity and are thus not heavily influenced when the number of elements is lowered by a factor.

In the settings that we usually mesh, these gains in time obtained by enforcing the combinatorial uniqueness of the elements in the queue are probably not worth the complexity of the implementation. However, one might expect that the difference would become more significant in a higher-dimensional setting, or for exceptionally large star sets.

4.4.4 Insertion of a point in the star set

A cycle can be broken down in four important steps: retrieving the next bad simplex to refine, computing the refinement point, inserting the refinement point in the star set and collecting the new bad simplices. The first two steps are described in Sections 4.4.3 and 4.2.1 respectively and the last one in Section 4.4.3. We focus in this section on the implementation details of the tasks performed after the refinement point is known. The two central tasks of the insertion of the point in the star set is the insertion of the point in existing stars, and the creation of the new star.

Efficiently inserting a new point in existing stars

Before describing our insertion algorithm, we first need to define a few notions related to the insertion of a point in a star.

A point p is said to be *in conflict* with some simplex σ of S_q if p is included in the G_q-circumball of σ. By extension, we say that a point is in conflict with a star if it is in conflict with one of its simplices. To ease the narration, we symmetrize the relation and say that a simplex (resp. star) is in conflict with a point if a point is in conflict with the simplex (resp. star). The *conflict zone* of S_q, denoted by Z_q, is the union of the G_q-circumballs $B_q(\sigma)$ of all the simplices σ of S_q. A point p is in conflict with a star S_q, if it belongs the conflict zone of S_q. By definition, a point p will belong to S_q after insertion in $\mathrm{Del}_q(\mathcal{P})$ if p is in Z_q. A star S_q for whom $p \in \mathrm{Z}_q$ is said to be *modified* by the insertion of p in $\mathrm{Del}_q(\mathcal{P})$.

Insertion in a stretched triangulation The insertion of a refinement point p in a triangulation $\mathrm{Del}_q(\mathcal{P})$ for $q \in \mathcal{P}$ and $q \neq p$ is thus performed by transforming the new point to the metric space and simply using traditional techniques such as the Bowyer-Watson algorithm [34, 138]. The insertion of a point is described in Algorithm 9.

Algorithm 9 Insertion of p in $\mathrm{Del}_q(\mathcal{P})$ with the Bowyer Watson algorithm

Compute the transformed point $T_p = F_q p$ (where F_q is the square root of G_q, the metric at q)
Compute the conflict zone of T_p in the Delaunay triangulation $F_q(\mathrm{Del}_q(\mathcal{P}))$
Delete all simplices that are in the conflict zone
Triangulate the hole with T_p, thus obtaining $F_q(\mathrm{Del}_q(\mathcal{P} \cup p))$

The insertion of a point in a star is illustrated in Figure 4.9.

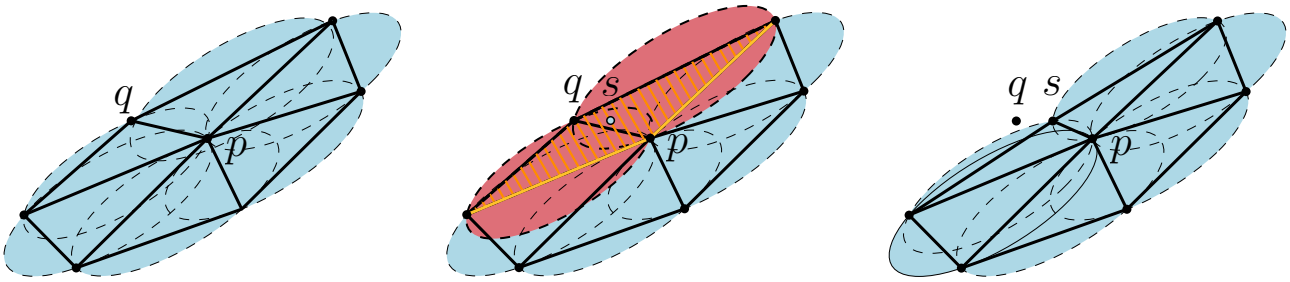

Figure 4.9: A star, its conflict zone (light blue), the refinement point (s) is in conflict with 2 Delaunay balls (red). The cavity computed during the Bowyer-Watson is dashed in orange with its borders in yellow.

In practice, we store the Delaunay triangulation at each point in its metric space (that is we store and maintain $F_q(\mathrm{Del}_q(\mathcal{P}))$) since almost all operations are easier to perform in the metric space, where $\mathrm{Del}_q(\mathcal{P})$ is an isotropic triangulation. For efficiency and memory reasons, we do not store the triangulation $\mathrm{Del}_q(\mathcal{P})$ and only cache the position of the center of the star in Euclidean space, which is sufficient.

Insertion in the star set A naive approach would insert a new point p in all the existing triangulations Del_q, with $q \in \mathcal{P}$. As the number of stars grows, this method quickly becomes too expensive and determining beforehand the stars $\{\mathrm{S}_{q_i}\}$ for whom p belongs to S_q after the insertion of p in $\mathrm{Del}_q(\mathcal{P})$ is thus essential to obtain an efficient implementation.

We can observe that a point p will belong in S_q only if p is in conflict with the simplices of S_q. Rather than inserting the point p in $\mathrm{Del}_q(\mathcal{P})$, we can thus only test if a point belongs to the conflict zone of a star and only insert the point in $\mathrm{Del}_q(\mathcal{P})$ if it belongs to Z_q, thus preventing many meaningless insertions. However, unions of spheres are difficult to handle swiftly, and traversing the full star set structure with each insertion is still not acceptable complexity-wise. To obtain an efficient detection of the conflicted stars, we use a bounding box-based filter: an AABB tree.

AABB tree To each conflict zone Z_q, we associate its Euclidean axis-parallel bounding box that we call simply the *bounding box* of q and denote by B_p. The bounding boxes of the current set of vertices $\mathcal{P}$ are maintained in a bounding volume hierarchy, an AABB tree [7]. Figure 4.10 shows the bounding boxes of a planar star and of a surface star.

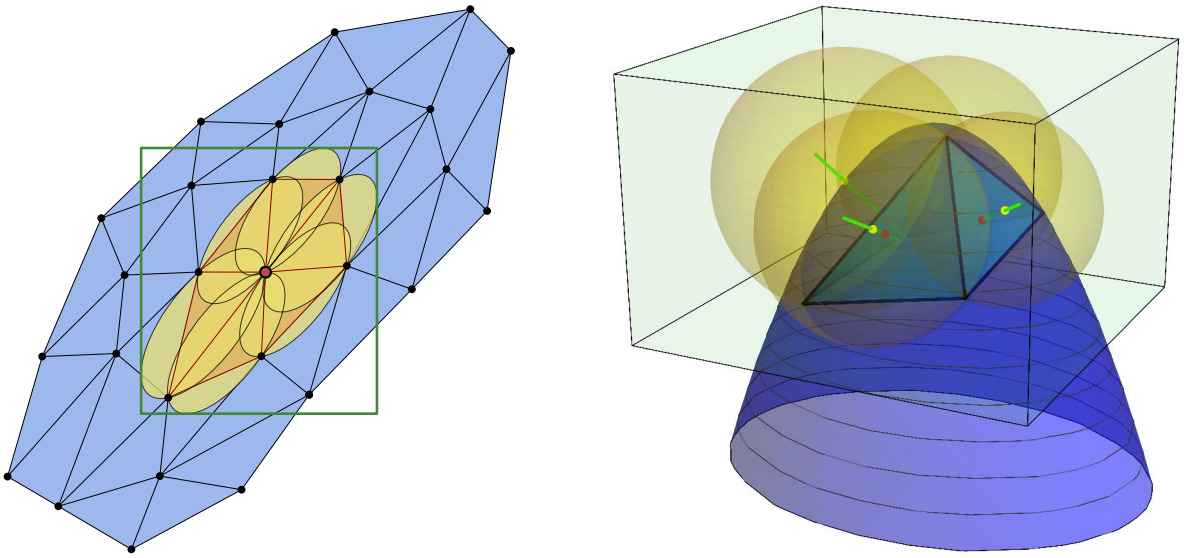

Figure 4.10: In blue, the mesh and the domain. In yellow, the Delaunay balls. In green, the box. On the right, yellow points are centers of the surface Delaunay balls and red points are facet circumcenters.

When inserting a new site p, we first query the AABB data structure and report all the boxes that contain p. If p belongs to some box B_q, we further check whether p belongs to the conflict zone Z_q and, in the affirmative, we insert p in the triangulation S_q.

A small issue arises from the fact that the bounding box of a star does not always decrease when a point is inserted in the star, see for example Figure 4.11. When this configuration is detected, the AABB tree must be rebalanced. Since this is a $O(n \log n)$ operation (with n the number of entries in the tree), it might seem problematic. However, this is a very rare phenomenon that only appears when the domain is not yet well approximated and consequently does not occur when the number of vertices is large.

Creation of a new star

The last step of a cycle is to create the new star S_p. The naive idea of building the complete triangulation $\mathrm{Del}_p(\mathcal{P})$ is once again computationally unreasonable. We must therefore find all points in $\mathcal{P}$ that will belong to S_p, but not (much) more. The stars $\{\mathrm{S}_{q_i}\}$ that are modified by the insertion of p provide a good starting set of points that are likely to belong to S_p: $\{q_i\}$. However, since the metric

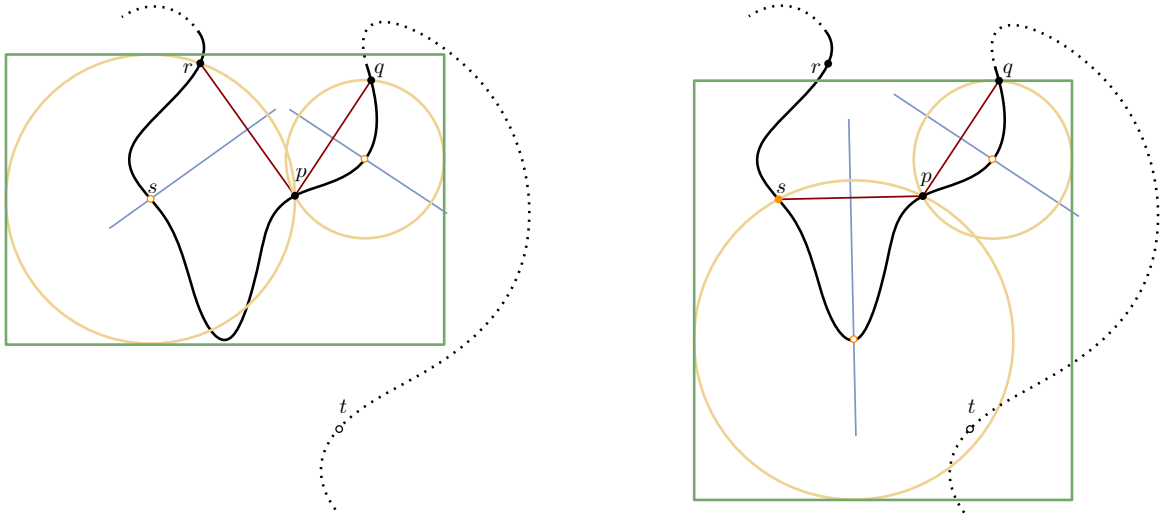

Figure 4.11: The box of S_p is growing after the insertion of s. The point t is an imaginary troublesome point used in Section 4.4.5.

field is generally non-uniform, there might be other points $\{r_i\}$ whose star was not modified by the insertion of p but still belong to S_p. Another filter, a k-d tree, is used to find these points efficiently.

k-d tree The current point set $\mathcal{P}$ is itself maintained in a k-d tree $\mathcal{K}d(V)$ [14]. While the AABB provides the knowledge of boxes containing a point, the k-d tree allows to quickly find which points are in a rectangular query.

The triangulation $\mathrm{Del}_p(\mathcal{P})$ of the new site is initialized as the star of p in the Delaunay triangulation of p and of all the vertices q_i whose star $S_{q_i}(\mathcal{P})$ has been modified by the insertion of p. We then compute the bounding box B_p of the current conflict zone Z_p, and query the k-d tree $\mathcal{K}d(V)$ to collect the set of vertices $\{r_i\}$ that fall within the bounding box. Each point r_i is inserted in $\mathrm{Del}_p(\mathcal{P})$ if it belongs to the current conflict zone Z_p.

Note that the insertion of a vertex modifies Z_p, thus there might exist a point s_i in conflict with S_p after the insertion of a point r_i, despite not being in conflict with S_p before insertion of r_i. This configuration is illustrated in Figure 4.11, where the point t is not in conflict with S_p until s is inserted in S_p. Thus, after each insertion, the conflict zone Z_p and the bounding box B_p must be updated, and the k-d tree queried again. This must repeated until there are no more vertices in conflict with Z_p. Individually cheap, this check is very costly over the whole meshing process. For example, in the generation of a mesh of 7000 vertices for a square endowed with a hyperbolic shock metric field, 10 seconds out of a total of 31 seconds are devoted to this check. The running time of this check naturally depends on how often the star set is cleaned (see Section 4.4.5).

Cavity caching

The boundary and internal simplices of the cavity C_q are used multiple times each cycle: when checking if a star is modified by the insertion of a point, when inserting a new point in a star, and finally when collecting elements that must be tested against the criteria. The cavity of a star is thus cached throughout a cycle.

4.4.5 Cleaning the star set

Storing all the triangulations of all the vertices would be costly from a memory point of view: if we stored all the triangulations $\mathrm{Del}_p(\mathcal{P})$, each point would appear as many times as there are points in the star set; and from a computational point of view: some operations, for example finding the location of a point, take longer in large Delaunay triangulations. To merge local triangulations $\mathrm{Del}_p(\mathcal{P})$ in a consistent triangulation of the domain, we thankfully do not actually need to maintain $\mathrm{Del}_p(\mathcal{P})$ for all $q \in \mathcal{P}$, but simply $\mathrm{S}_p(\mathcal{P})$, which is a relatively constant number of vertices since we are using a Delaunay refinement algorithm.

In practice, we store for each $p \in \mathcal{P}$ a triangulation $\mathrm{Del}'_p(\mathcal{P})$ that is kept close to S_p by trimming every n_c (this value is discussed below) the vertices that are not incident to p, which happens as we insert new vertices in $\mathrm{Del}'_p(\mathcal{P})$, a process referred to as *cleaning*. To not have to seek which stars must be cleaned, we keep in memory a list of stars that have been modified since the previous time stars were cleaned.

The value n_c naturally influences the runtime of the algorithm (but not its result) and must thus be carefully picked: when n_c is chosen too small, the cost of regularly rebuilding stars outweighs the benefit of handling small triangulations; the opposite happens when n_c is too large. Table 4.1 shows the effect on the runtime for different values of n_c.

Table 4.1: Influence of cleaning on the computational time. The final mesh has 22731 vertices.

Cleaning every	1	5	10	50	75	100
Time (s)	617.86	593.80	544.23	556.49	572.33	588.49
Cleaning every	250	500	1000	5000	10000	20000
Time (s)	595.14	599.39	615.86	648.98	705.17	1060.69

These results are relatively consistent over all our results and we thus use $n_c = 50$, which provides a good balance between both the cost of cleaning and the size of triangulations. Note that except for the degenerate case of a point set with a size smaller than n_c, this value is independent of the total size of the mesh as it compares the cost of rebuilding a star (which is a completely local operation if we assume that we keep correct stars at all times) of stars versus the cost of inserting in a local neighborhood.

4.4.6 Parallelization

At first glance, the star structure of the algorithm might seem perfect for parallelization. In practice, it is more difficult: simply parallelizing the insertions of a point in each of the conflicted stars – even including finding the new bad simplices and their insertions in the refinement queues of the new bad simplices – will actually result in a slow down of the algorithm, due to cache issues. Indeed, the class `Stretched_Delaunay` has a heavy memory footprint caused by a large amount of members: a Delaunay triangulation, a metric, multiple combinatorial information caching structures, filters (see Section 4.4.4), etc.

Another option would be to parallelize the algorithm in a similar fashion to the isotropic mesh generator Mesh_3 [80], with each thread handling the refinement of a single simplex at a time. In the parallel version of Mesh_3, the principle of *locks* are used to avoid refining simplices that are too close from each other. The implementation is however technical and tedious, and the anisotropy of stars makes the locality of locks more difficult to handle. While this technique would potentially

accelerate the refinement process, it was not pursued due to other unrelated issues with the algorithm (see Section 4.7).

A parallel approach, based on surface segmentation, is nevertheless investigated in Chapter III.6 where we will need to build inconsistent star sets swiftly.

4.5 Discussion on the parameters

The algorithm relies on various parameters: ψ_0, r_0, ρ_0, σ_0, β, δ... It can be difficult, at first glance, to estimate their influence on the outcome in term of speed, number of vertices or quality of the produced mesh. We investigate each parameter independently.

4.5.1 Parameter ψ_0

The rule with the highest priority in the algorithm is the distortion rule, which bounds the maximal distortion in a simplex to be at most ψ_0, with the intent of increasing the odds of finding a valid solution when the `Pick_valid` procedure is called later in the algorithm. The value ψ_0 has naturally a strong impact over both the computation time and the final mesh: if ψ_0 is chosen too large, the `Pick_valid` procedure will be at first largely unsuccessful, causing many unsuccessful and costly insertions till the sampling increases and valid points are be found. Oppositely, if ψ_0 is chosen too small, simplices might be refined even if they already satisfy all other criteria – including consistency – which is pointless and costly and thus undesirable.

To determine what is the optimal value for ψ_0, we look at the number of vertices in a final mesh, the number of calls to the `Pick_valid` procedure and their success percentage, and the computation time for various values of ψ_0. All the parameters except for ψ_0 are kept constant and the sizing constraint r_0 is chosen large enough as to only be responsible for a negligible number of vertices added. The domain is a square of side 4 and centered on $c = (0, 0)$ endowed with the hyperbolic shock (detailed in Section A.3 in Appendix A). Results are shown in Figure 4.12.

For the sake of completeness, we also consider a three-dimensional domain, a cube of side 3 and centered on $c = \{1.5, 1.5, 1.5\}$, endowed with a unidimensional shock metric field (see Section A.8 in Appendix A). Since the goal is to compare the influence of ψ_0 for the refinement of cells, we disable the refinement of surface stars and construct a pure volume mesh. Results can be found in Figure 4.13.

These experiments provide a lot of interesting and unexpected information:

- The success percentage of the `Pick_valid` procedure increases when ψ_0 decreases, as the theory predicts. However, a low value of ψ_0 will very quickly be responsible for the insertion of an exceptionally large number of vertices (red curves).

- The final number of points is greater whenever ψ_0 is used than when it is not (blue curves).

- However small ψ_0 is, there are still inconsistencies that appear (the teal curves show the number of vertices inserted to solve inconsistencies), proving that the `Pick_valid` procedure is a necessary part of the algorithm. Even more interestingly, they add a relatively constant number of vertices, indicating that the distortion rules did not help.

- Despite Rule 1 inserting only the circumcenter, for no value of ψ_0 is time gained by using the distortion criterion: the mesh was generated fastest when no simplex was refined for the distortion (green curves).

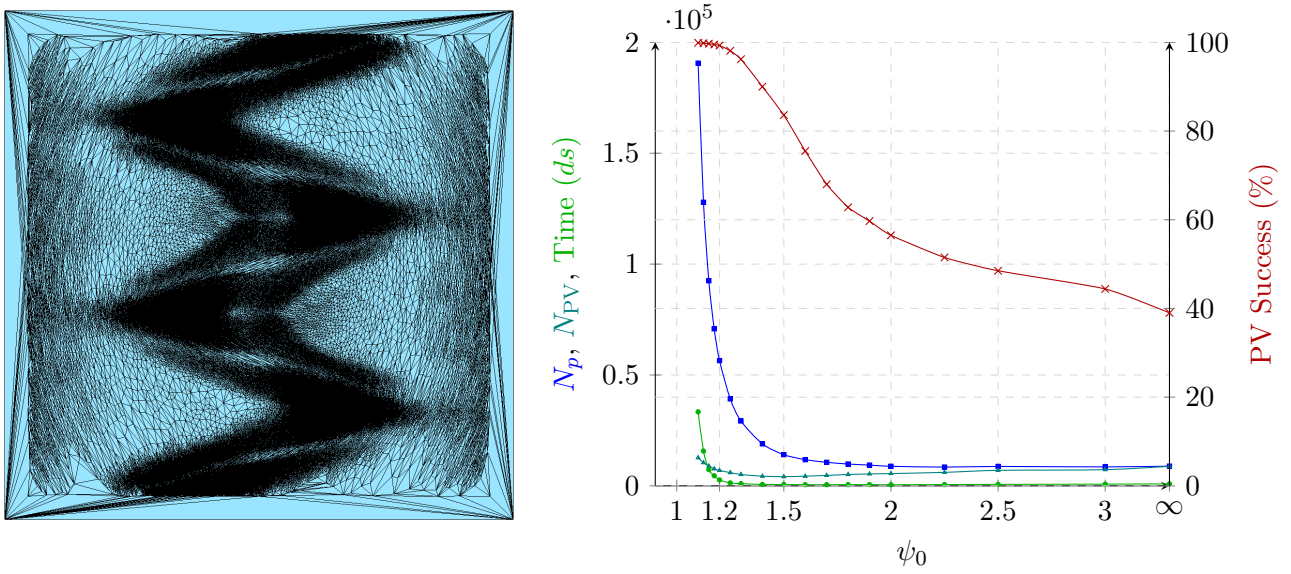

Figure 4.12: Influence of the parameter ψ_0 in a two-dimensional domain (shown on the left). In blue, the final number of vertices (N_p). In teal, the number of calls to the `Pick_valid` procedure (N_{PV}). In red, the percentage of success of the `Pick_valid` procedure. In green, the time (in deciseconds). The value ∞ means that the distortion queue is disabled.

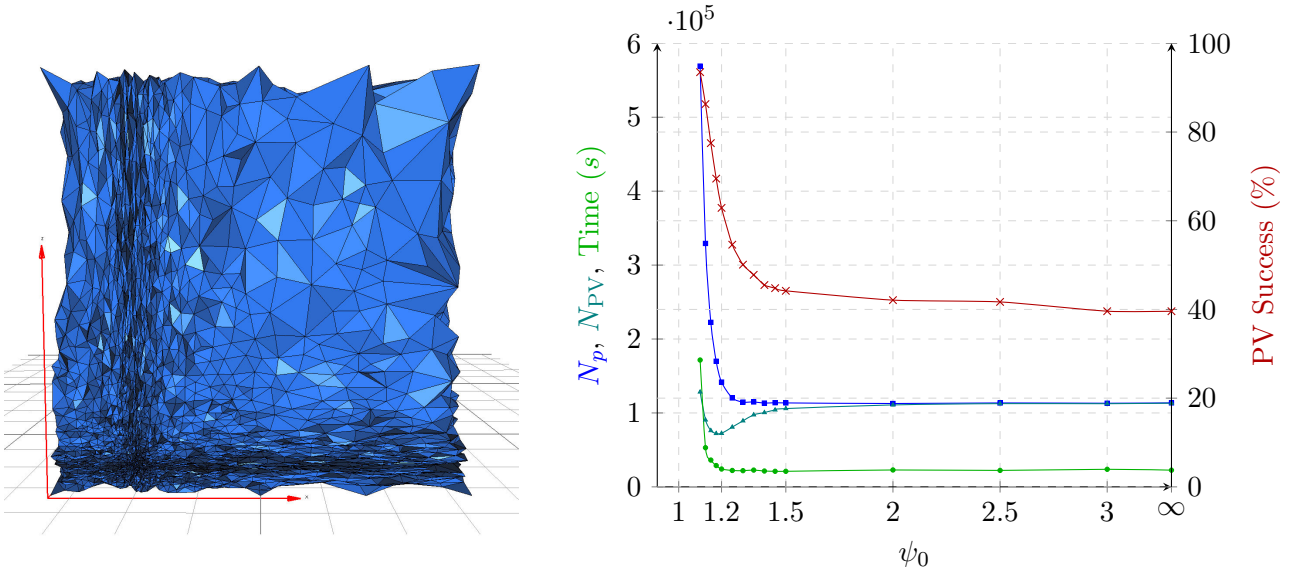

Figure 4.13: Influence of the parameter ψ_0 in a three-dimensional domain (shown on the left). In blue, the final number of vertices (N_p). In teal, the number of calls to the `Pick_valid` procedure (N_{PV}). In red, the percentage of success of the `Pick_valid` procedure. In green, the time (in seconds). The value ∞ means that the distortion queue is disabled.

Additionally, the comparison between Figures 4.12 and 4.13 shows that while the profiles of all curves are similar in 2 and 3D, the bound below which ψ_0 is responsible for a large increase in the number

of vertices in the final mesh is different, making it difficult to choose a value of ψ_0 if the Rule 1 s beneficial.

In conclusion, the distortion queue is – in practice – a rule that is at best not useful, and at worst extremely harmful to the algorithm. We shall modify the general algorithm to take into account these observations, see Section 4.6.1.

4.5.2 Parameters r_0 and ρ_0

The values of r_0 (size criterion) and ρ_0 (shape criterion) naturally affect the number of vertices as additional vertices must be inserted to satisfy these criteria. In isotropic mesh generation, the number of vertices follows roughly the square of the sizing field in 2D and the cube in 3D: if the sizing field is divided by 2, there are 4 times more vertices in 2D and 8 times more vertices in 3D.

As explained in Section 2.13 in Chapter 2 and in Section 4.2.2, the size parameter is usually left at 1 as the size of the elements can be directly encoded in the metric. We exceptionally scale r_0 instead of the metric field in this experiment for the purpose of clarity. The results are shown in Figure 4.14.

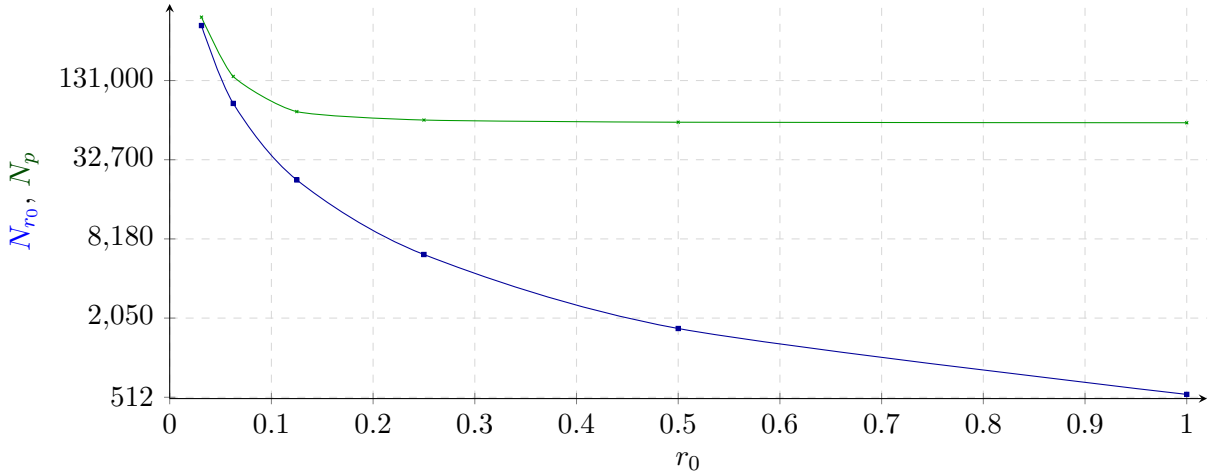

Figure 4.14: Influence of the parameter r_0 on the final number of vertices. N_{r_0} is the number of points added by the sizing criterion. N_p is the total number of points in the final mesh.

This relation is somewhat similar in the case of anisotropic mesh generation. Here again, the number of points added by the inconsistency queue (that can be roughly computed as the difference of the green and blue curve) is more or less constant, even when the number of points inserted for the size rule is important.

The shape parameter ρ_0 and its corresponding rule ensure that all simplices satisfy shape requirements, which is both useful in itself but also improve the odds of finding valid solutions in the `Pick_valid` procedure. This parameter has, in practice, very little influence and the corresponding rule is very rarely called upon. This can be seen as a consequence from the fact that Rules 1 and 2 (distortion and size) insert the circumcenter, an already appropriate choice to create meshes of good quality (see Ruppert).

4.5.3 Parameters β and δ

Contrary to the parameters ψ_0 and ρ_0, who serve to satisfy quality requirements on simplices ahead of `Pick_valid` procedure calls, the parameters β and δ are directly involved in the `Pick_valid`

procedure. Indeed, the parameter β controls the size of acceptable inconsistencies and δ determines the size of the picking region $P_p(\tau)$ as its radius is given by $\delta r_p(\tau)$.

We use a cubic domain of side 1, centered on $(0.5, 0.5, 0.5)$ and endowed with a unidimensional shock metric field with maximal anisotropy 3 (see Section A.8 in Appendix A). The number of `Pick_valid` tries is fixed at 60 but the value of δ varies. We investigate the final number of vertices in the mesh and the quality of the simplices. For planar and surface domains, the quality is estimated with the formula of Zhong et al. [144]:

$$Q = 4\sqrt{3}\,\frac{A}{ph}.$$

where A is the area of the triangle, p the perimeter and h the longest edge (all computed in the metric). For cells, we use the quality estimation of Frey and George [71]:

$$Q = 216\sqrt{3}\,\frac{V^2}{A_\Sigma^3}.$$

where V is the volume of the tetrahedron, and A_Σ the sum of the areas of the four facets (all compute in the metric). Both these quality measures live between 0 and 1, with 1 signaling the highest quality. Results are detailed in Table 4.2.

Table 4.2: Comparison of the number of vertices and quality of the mesh for different values of δ.

δ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Points	1173	734	497	364	356	368	427	490	678	720
PV success	345	204	146	92	95	118	148	194	310	354
PV failures	635	386	231	161	158	156	177	178	234	228
Quality (min)	0.05	0.02	0.01	0.03	0.03	0.02	0.04	0.08	0.001	2×10^{-3}
Quality (max)	0.993	0.989	0.976	0.980	0.984	0.986	0.975	0.980	0.987	0.993
Quality (avg)	0.703	0.704	0.684	0.678	0.648	0.631	0.610	0.600	0.566	0.549

As expected, more solutions are available in the pick valid algorithm when the picking region is enlarged, resulting in smaller meshes. However, the metric is followed more loosely and the simplices of the meshes have lower quality. Note that in extreme cases – δ close to 1 – the final number of vertices starts to increase again, as the refinement points are often not creating satisfying elements with respect to other criteria, such as the shape.

The parameter β is assigned the value 2.5 by default, but changing this value (within 1 to 5) has barely any influence on the outcome.

4.5.4 Parameters σ_0

The parameter σ_0 controls the maximal sliverity of the simplices (as defined in Section 4.1.2). A majority of inconsistencies do not stem from slivery quasi-cosphericities, and thus this parameter has little influence on the outcome. By increasing its value, one simply trades the removal of inconsistencies for the removal of silvers, but the result stays the same. Furthermore, the slivery and inconsistent queue both rely on the `Pick_valid` procedure to insert a point and thus there is no difference in the running time of the algorithm either.

4.6 On the usage of `Pick_valid`

Once other criteria are satisfied, it is necessary to solve inconsistencies before the star set can be merged to form a triangulation. The `Pick_valid` procedure seeks Steiner points that can refine inconsistent simplices without creating new inconsistencies. It is proven both theoretically (see Boissonnat et al. [31]) and empirically (see Section 4.5.1) that the procedure is essential to the termination of the algorithm.

Estimating beforehand whether a valid Steiner point can be found by the `Pick_valid` procedure is difficult. Even if we knew that a valid candidate exists in the picking region, how should we proceed to find the solution quickly? These questions are important from a practical point of view as picking and testing a candidate point is an expensive process (as will be shown in this Section).

Additionally, the algorithm described in Algorithm 1 to test the validity of a candidate is rather theoretical: a valid candidate should not form a (small) quasi-cosphericity with any combination of n vertices in $\mathcal{P}$. Recall that a set of $n+1$ points (in dimension $\mathbb{R}^n$) defines a forbidden ring whose geometry is a thickened ellipsoid (or a thickened sphere, in the metric space); the intersection of a forbidden ring with the picking region is called a forbidden zone.

Remark 4.6.1 *Note that we have assumed that all criteria are now solved: the distortion is sufficiently low, and the slivers have been removed. Thus, all the inconsistencies in the mesh are created by quasi-cosphericities that do not involve any slivers.*

In practice, there are many possible combinations, with most of them creating a forbidden ring that does not intersect the picking region and is thus irrelevant. It is thus not reasonable to compute all of the forbidden rings. How should one test the validity of a candidate efficiently ?

Validity of a candidate

Collecting and computing all the quasi-cosphericities around a simplex is difficult and computationally expensive. We proceed instead in a naive approach: we simulate the insertion of a candidate in the star set and scan the newly simplices for bad cosphericities. In the presence of such bad configurations, the insertion is reverted, otherwise, the candidate is a valid point and the insertion has already been performed.

Consequently, testing a single candidate point has the complexity of inserting a point in the star set.

Search for valid candidates

In the description of the `Pick_valid` procedure, candidate points are randomly selected within the picking region. Using random points might not seem like the optimal choice since forbidden zones of the picking region are predictable geometries determined by neighboring vertices. Indeed, forbidden zones invalidate points along thickened ellipsoids in the picking region (see Figure 4.5) and one could hope to exploit this fact. Three possibilities were investigated to try to improve the speed of pick valid.

Analytical approach A first approach to improve the selection of candidates is entirely analytical and based on the following reasoning: if we could compute closed forms for both the forbidden rings and of the picking region, we could then intersect these regions to determine if a valid candidate exists, and where.

Placing ourselves in the metric, the circumscribing balls of simplices is a sphere, hence the picking region is a sphere and the quasi-cosphericities are thickened spheres (whose thickness can be bounded relatively tightly). It is thus easy to compute each region independently. However, forbidden rings are created by any combination of $n+1$ vertices in the neighboring vertices around the simplex being refined (including the vertices of the simplex itself); there is often a large number of forbidden rings that intersect the picking region. Consequently, we must compute the intersection of multiple spheres. It is not easy to do so, and even less to perform it on-the-fly and fast.

A rough evaluation of these sizes is computed in the proof of termination of the algorithm which gives an upper bound on this volume. Unfortunately the bound simply sums the volumes of the thickened spheres of each quasi-cosphericity and this is far too rough to be used in practice.

The analytical approach thus cannot be used efficiently.

Griding the picking region To avoid complicated calculations, a second approach is based on the discretization of the picking region with either a grid or a triangulation. When a candidate point is found to be invalid, the forbidden zone that invalidated the candidate is computed and all the vertices of the grid that fall within the forbidden zone are also invalidated. The next candidate is then picked from the set of remaining non-invalidated points. The discretized pick valid region of a simplex is illustrated in Figure 4.15.

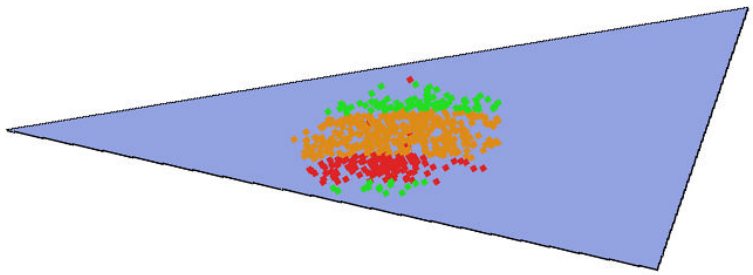

Figure 4.15: Discrete point set of the pick region of a simplex. Invalid points are marked in red and orange, and valid points in green. The geometry of the two forbidden rings is clearly distinguishable.

This approach performs better than our first idea but is not practical in the end: either few forbidden rings intersect the picking region and a valid point will be rapidly found with random draws, or there are only few solutions and the discretization needs to be dense. In the latter case, the computational cost is shifted from testing points to generating and maintaining the grid; overall, no time is gained compared to a random approach.

Random selection with memory The last approach tried to capitalize once again on the predetermined geometry (a sphere, in the metric space) of forbidden rings to pick candidates. In the event of a random point p not being valid, the next random point p' is picked by moving away from the center(s) of the forbidden ring(s) that invalidated p with the idea that we move away from the thickened sphere that is the quasi-cosphericity.

Estimating a direction using the last k forbidden zones (for variable k) was tested, but this was not effective in practice for similar reasons as the second method. The number of forbidden zones is again the culprit of the impracticality of this approach: if there are only few forbidden rings, a valid random point is likely to be found quickly; but if there is a large number of forbidden zones, it is difficult to take them all into account as we decide of the direction towards which to move after an invalid point.

In conclusion, it is simply easier and quicker to try random points and test whether they are valid or not. However, since there is not necessarily a solution, the number of tries has to be limited. It is not necessarily clear how many times should one try, and we investigate this question next.

Generation of random points The generation of a random point in the picking region of a simplex of a volume star is performed using the geometric object generator package of CGAL. The picking region of a surface facet τ is defined as the intersection of the manifold $\mathcal{M}$ that bounds the domain and a scaled down version $\mathcal{B}'(\tau)$ of the surface Delaunay ball of the facet. To generate points in this region, we pick two random points in $\mathcal{B}_s$ (see Figure 4.6) and intersect the segment formed by joining these two points with $\mathcal{M}$.

Number of random tries

Since random tries turn out to be the most convenient approach and knowing that there is not always a valid solution, we must set a maximum number of tries that we should attempt lest we get stuck in the procedure. As no clear value can be guessed, we study the influence of the number of tries on the success of the procedure, and the effects on the final mesh. For consistency, we consider the same input as in previous experiments, a square centered on 0 and of side 1.5 endowed with the hyperbolic metric field (described in Section A.3 of Appendix A). The distortion queue is disabled and the sizing criterion is here again made irrelevant; (almost) every insertion thus requires the `Pick_valid` procedure. Figure 4.16 shows the results in terms of final number of vertices and computational time for two choices of the maximal number of tries.

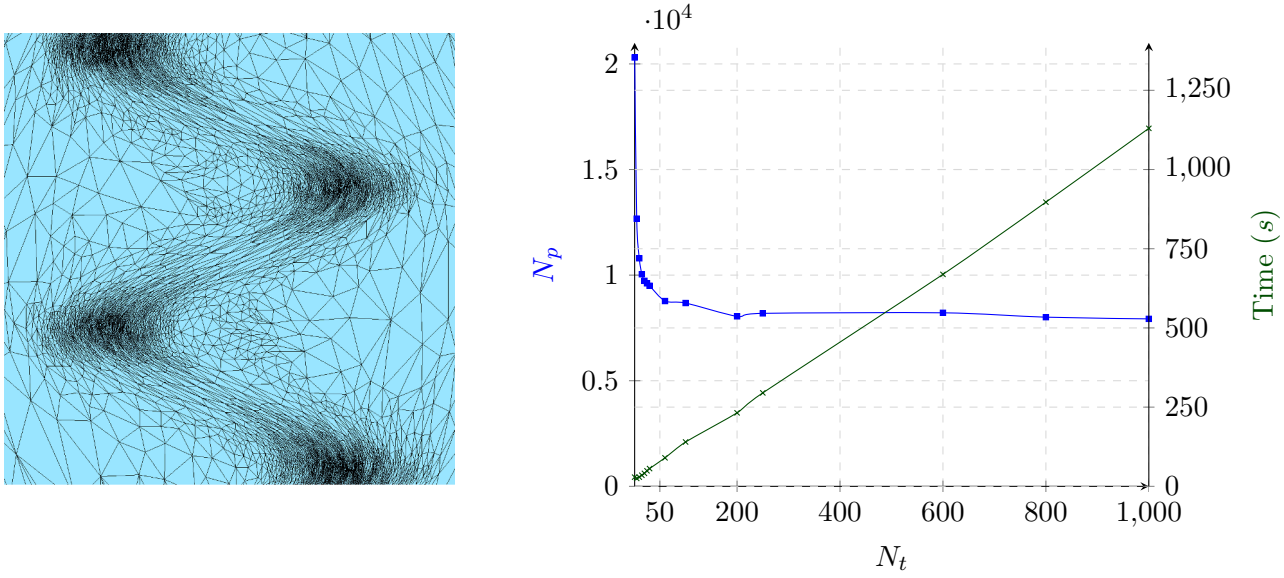

Figure 4.16: Influence of the parameter N_t over the final number of vertices and the computational time. The domain is a square of side 3, endowed with the hyperbolic shock (the mesh is shown on the left). The blue curve shows the final number of vertices and the green curve the time, in seconds.

From the blue curve, we can conclude that the number of vertices required to obtain consistency significantly drops when the amount of tries grows, but only up to a certain point. As a side note, we can observe that the computational time is almost linear with respect to the number of tries (green

curve), which confirms that the picking and testing a candidate point in the Pick_valid algorithm is thus as expensive as the insertion of a point. This is not surprising since the testing for validity is literally the insertion of the candidate in the star set. It shows however that all other maintenance related functions have negligible cost compared to the insertion of a vertex in the star set.

In accordance with the result of this experiment – other inputs provide similar results – the maximal number of tries is set to 60.

4.6.1 Improvements to the Pick_valid procedure

As shown by experiments in Section 4.6, the Pick_valid procedure is, from a runtime point of view, the clear bottleneck of the algorithm. Minor modifications were made to the Pick_valid procedure to minimize the amount of calls to this procedure and increase the speed of the test of a candidate's validity. These changes affect both the runtime of the procedure, but also its success rate. The first two modifications are used to avoid calling the Pick_valid procedure if it is unlikely to find a valid point, or if its result is irrelevant, thus preventing useless and expensive computations. The third improvement acts on the order of simplices in the refinement queues, and aims to increase the speed by reducing the number of vertices in the final mesh.

Distortion skip in the Pick_valid procedure

In Section 4.5.1, we have proved that using a distortion criterion was in fact harmful to the algorithm. Nevertheless, we know that in theory the Pick_valid procedure is more likely to fail for large distortions. To measure this success probability, we disable the distortion criterion and observe the outcome of the Pick_valid procedure during a single Delaunay refinement run. In particular, we study the likelihood of the Pick_valid procedure to succeed with respect to the distortion of the simplex being refined. Note that this is slightly different from Figures 4.12 and 4.13 as the distortion queue was then used and ensured that the distortion within all simplices was below a bound, which is not necessarily the case here.

From Figure 4.17, we observe that – as expected – the Pick_valid procedure is unlikely to find valid points for large distortions. A lower percentage of success compared to Figures 4.12 and 4.13 can be explained by the fact that the Pick_valid procedure heavily depends on vertices that neighbor the simplex and that the distortion between all the neighboring vertices might be larger than the distortion of the simplex.

To take into account the results of this experiment in our algorithm, we introduce a "skip" based on the distortion in the Pick_valid procedure: if a simplex $\sigma \in \mathrm{S}_p$ must be refined through the Pick_valid procedure but its distortion $\psi(\sigma)$ is larger than a chosen bound ψ_0, then σ is simply destroyed by inserting its circumcenter $c_p(\sigma)$. While it may seem very similar to the distortion rule that was discarded, this skip is a much better version as it only applies within the Pick_valid procedure and will thus not over-refine the mesh. The improved Pick_valid procedure is presented in Algorithm 10.

The algorithm avoids unnecessary calls to the Pick_valid procedure without the downsides of a distortion queues. Figure 4.18 shows the results obtained with this skip for various values of ψ_0; ∞ means that the distortion skip was not used. The modification has a non negligible cost, which can be seen by the slight increase in computational time when activating the skip but using a very large ψ_0 (10). However, the skip then allows for increasing gains of speed. As for the original distortion queue (see Section 4.5.1), a value of ψ_0 chosen too low creates a large number of points that are not otherwise required. A small enough bound ψ_0 will eventually cause the cost of adding superfluous points balance to increase and reduce the usefulness of the skip, which can be seen on for values

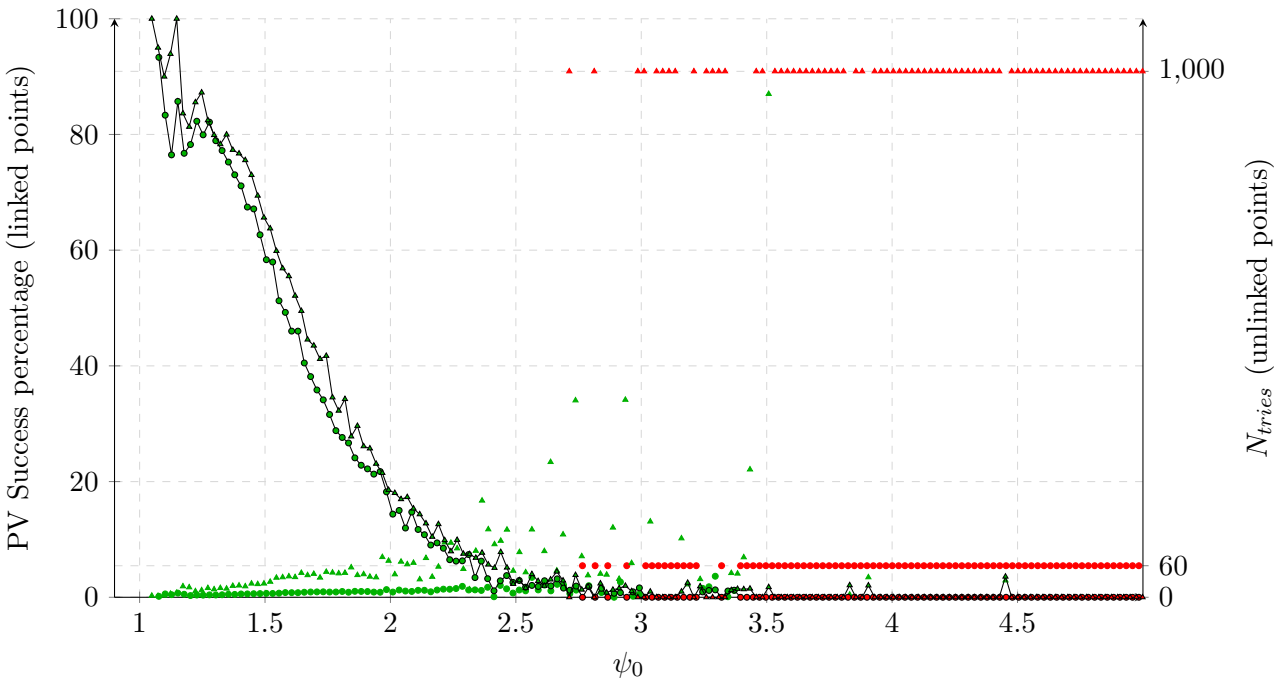

Figure 4.17: Success of the `Pick_valid` procedure for 60 and 1000 tries. On the graph, circular entries are obtained using $N_{tries} = 60$, and triangular entries are obtained using $N_{tries} = 1000$. Entries referring to the percentage are linked together with black segments. Unlinked entries indicate how many tries were needed on average for success. A green entry indicates that the `Pick_valid` procedure usually succeeded, while a red entry means that the procedure almost always failed.

Algorithm 10 Improved `Pick_valid` procedure

```
if ψ(τ) < ψ0 then
    while No valid point has been found do
        Pick randomly a point p in the picking region P_p(τ)
        if p is valid then                          ▷ Checked as in Algorithm 1
            Return p
else
    Return c_p(τ)
```

$\psi_0 < 1.3$ on the graph. In practice, we use $\psi_0 = 2$ as it reduces significantly the computational time without increasing the number of vertices.

Encroachment skip in Pick_valid

In Section 4.3.3, we have described how a refinement point that encroaches a lower mesher level is discarded, and the lower mesher level is instead called to refine the simplex whose Delaunay ball the refinement fell in. Checking whether a candidate encroaches a lower mesher level before testing its validity is another small change that can avoid costly calls to the `Pick_valid` procedure. This change is obviously beneficial but the extent of the gain depends on too many criteria (approximation error, sizing of facets and cells, domain) to produce some meaningful comparison.

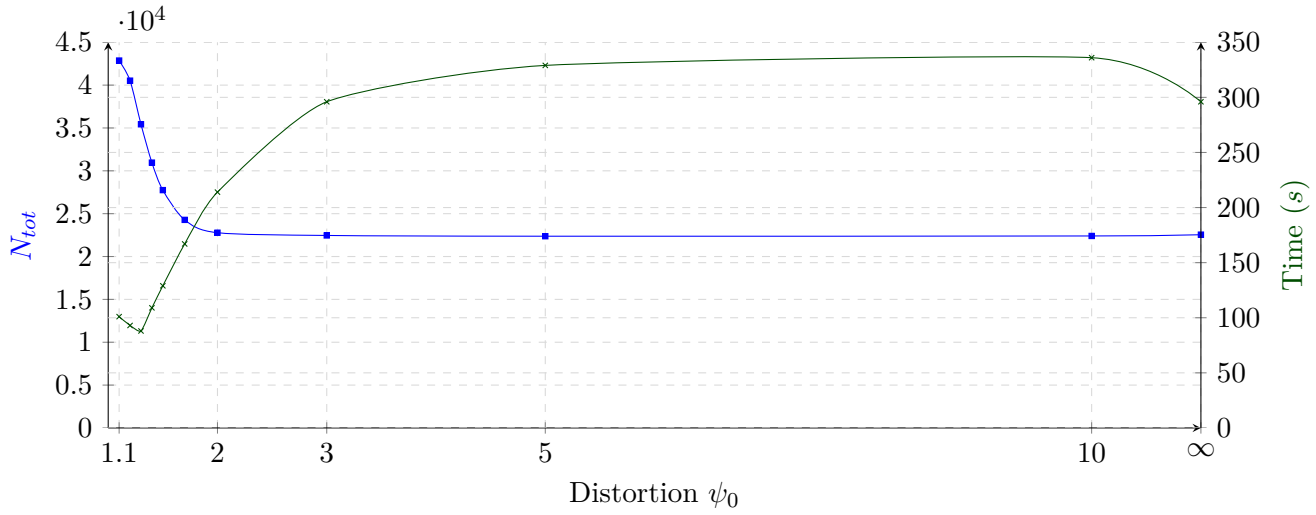

Figure 4.18: Influence of the distortion skip for various values of ψ_0.

A related but rougher change is to completely bypass the `Pick_valid` procedure if the center of the picking region encroaches a lower mesher level, and to simply call the `Pick_valid` procedure directly at the lower mesher level. It is qualified of "rougher" because in theory – and this is confirmed in practice – it might be possible to find non-encroaching valid points in the picking region even if the center of the picking region encroaches a facet Using this "skip" is a way to trade a larger number of vertices for a lower running time.

Temporary rejection in the refinement queues

In the original algorithm, the elements are put in priority queues and sorted such that the worst element of a queue is treated first. In the case of the inconsistency queue, the volume of simplices is used to order elements: the larger the volume, the higher the priority. We focus here on queues that use the `Pick_valid` procedure to find a refinement point: shape, sliverity and inconsistency rules. From previous experiences, we see that failures in the `Pick_valid` procedure are costly as they often mean that we end up adding a point – the circumcenter – which creates inconsistencies and we try again on a new simplex (with slightly better odds).

We have seen that increasing the number of tries helps reducing the number of points (see Section 4.6). However, if the maximum number of tries N_t is fixed, could simply delaying the refinement of an element for which the `Pick_valid` procedure failed (instead of adding the circumcenter) and considering the next bad simplex improve the number of points? Indeed, there might exist bad simplices for which the `Pick_valid` procedure succeeds and the insertion of the (valid) point also destroys a simplex for which we had no solution.

We investigate the effect of delaying the refinement of a simplex for which the `Pick_valid` procedure does not find a valid point. The bad simplex is moved to the back of the queue by artificially changing its badness score (to that of the last element minus a small value). Moving an element to the bottom of the queue is fast: removal and insertion in the multiset can here be done with hints, thus resulting in average constant complexity. To avoid cyclical failures, the rejection of a simplex to the back of the queue is limited to a maximum of a single rejection.

The final `Pick_valid` procedure that uses all the modifications of these three sections is presented

in Algorithm 11.

Algorithm 11 Final `Pick_valid` algorithm

Let $number\_of\_tries = 0$
if $\psi(\tau) > \psi_0$ **then** Return $c_p(\tau)$
while $number\_of\_tries < N_t$ **do**
  Pick randomly a point p in the picking region $\mathrm{P}_p(\tau)$
  if p encroaches a lower mesher level **then**
    Continue
  if p is valid **then**
    Return p.
if τ is not the last element in the queue AND τ has not already failed a pick valid test **then**
  Move τ to the bottom of the queue
  Try to refine the next element of the queue
else
  Return $c_p(\tau)$

Comparison

We analyze the improvements brought by our modified `Pick_valid` procedure by comparing it with the former algorithm. We consider the cube of side $c = 1$, centered on the $(0.5, 0.5, 0.5)$ and all the default parameters. Table 4.3 shows the difference between the two `Pick_valid` procedures depending on the amount of tries we fixed. For each procedure, the table lists the final number of vertices in the mesh, the number of calls to the procedure, the time spent in the procedure and the time spent updating the queue.

Table 4.3: Effectiveness of the modified `Pick_valid` algorithm. "OPV" stands for Origin `Pick_valid` and "NPV" for New `Pick_valid`.

Tries	1	15	30	60	125	250	500	1000	2000
Points (OPV)	6396	1594	1167	823	640	575	452	481	397
Calls of PV	6380	1578	1151	807	624	559	436	465	381
Time in PV	59.38	32.18	34.03	36.624	44.11	80.69	117.32	244.11	370.80
Total Time	154.20	42.173	40.84	41.06	46.58	82.81	118.84	245.76	372.12
Points (NPV)	1774	585	562	452	391	319	338	234	245
Calls of PV	20730	4933	4607	3606	3144	2552	2940	1636	1842
Rejections	14972	4364	4061	3170	2767	2249	2618	1418	1612
Time in PV	46.96	65.773	120.47	183.49	317.82	499.7	1144.43	1166.25	1216.3
Time Updating	0.02	0.01	0.01	0.01	0.01	0	0	0	0
Total Time	61.14	67.98	122.68	185.41	319.042	500.7	1145.48	1166.81	1216.7

For a constant amount of tries, the rejection trick results in a large number of additional calls to the `Pick_valid` procedure, and thus a longer run time. However, the number of vertices is significantly cut. While this does not help improving the speed of the algorithm, it is nevertheless interesting that

this trick can help reduce the number of vertices in the final mesh, similarly to an increase of the maximal number of tries in the `Pick_valid` procedure.

4.7 Results and limitations

We now present meshes produced by our implementation. Domains and metrics fields are picked as diversely as possible to explore the full capabilities of the algorithm. Details on inputs can be found in Appendices A for metrics and B for domains respectively. Unless explicatively stated otherwise, the parameters are set to their default value: $\psi_0 = \infty$ (no distortion refinement), $r_0 = 1$ (for faces, facets, or cells), $\rho_0 = 3$, $\beta = 2.5$ and $\delta = 0.3$. The `Pick_valid` procedure uses a maximum of 60 random tries, and uses all its improvements.

Remark 4.7.1 *When aiming to build unit meshes (see Section 2.13 in Chapter 2) with a fixed tolerance (say $\sqrt{2}$), not every metric field is physically possible. We always choose carefully the sizing field parameter of our metric fields such that it is physically possible to construct a unit mesh of the input.*

The results are presented in this section with only light commentary; we will provide a deeper analysis in Section 4.8. A table of in-depth statistics for all results is provided in Section 4.7.6 and the speed of the performance of our implementation is described in Section 4.7.7.

4.7.1 Uniform metric fields

A uniform metric field associates to any point of the domain the same metric. Note that since all the metrics are identical, neighboring stars necessarily have compatible connectivities and there can be no inconsistencies. It should be noted that in the setting of uniform metric fields, an anisotropic mesh can simply be obtained by stretching the domain through the usual metric transformation (the square root of the metric), meshing isotropically in the metric space and finally stretching back the constructed isotropic Delaunay triangulation to obtain an anisotropic Delaunay triangulation – which is in fact what is done in each star, but for different metrics in the general case.

This setting is handled without any issue by the algorithm and our implementation, whether in the setting of planar, surface or volume domains: the algorithm terminates, all criteria are honored and we obtain edges that have a length close to 1 (see Section 4.7.6). As the algorithm perform most computations in the metric space – where triangulations are isotropic – the algorithm is extremely robust and can handle anisotropy ratios up to 10^{17}; numerical issues appear in the transformations for higher ratios, which could easily be fixed by changing number types (but no one realistically deals with such ratios).

4.7.2 Shock-based metric fields on planar domains

We depart from uniform metric fields and consider a planar domain – a square – endowed with various metric fields. These artificial metric fields are chosen to exhibit different phenomena:

- Straight shock — A region of high-anisotropy along a relatively constant direction in between two regions of lower anisotropy. This corresponds to a change in the eigenvalues of the metric. Waves along a (straight) shore is a good illustration of such phenomenon.
- Rotational anisotropy — A region where the anisotropy ratio is constant, but the direction of the stretching is changing. This corresponds to a change in the eigenvectors of the metric. We will sometimes say that the metric is *rotating* to describe such regions.

These two phenomena are the primal blocks and can be combined and scaled to form any more complex or "real-world" metric fields.

Starred

Our first example uses the "Starred" metric field, detailed in Section A.5 of Appendix A. The anisotropy of this metric field varies between 1 and 10. The "Starred" metric field possesses long – compared to the prescribed size of the elements – regions of straight anisotropy, and regions where the metric field is rotating (and the anisotropy ratio is lower) in between, thus providing a good first example of an arbitrary metric field. The domain is a square of side 10, centered on the origin.

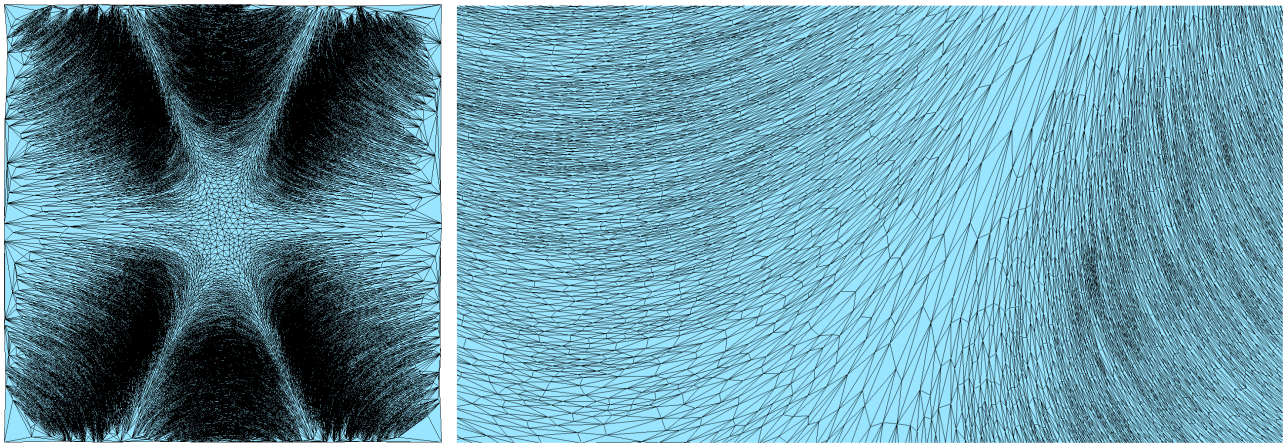

Figure 4.19: A square of side 10 and centered on the origin, endowed with the "Starred" metric field (left). The final mesh is composed of 47126 vertices and 94366 triangles. On the right, a zoom on one of the rotating regions.

The algorithm terminates without any issue and the size and shape criteria are solved quickly (4499 vertices are needed). However, the resolution of inconsistencies is difficult and requires around 40000 additional vertices. While regions where the metric field is rotating are clearly suffering from inconsistencies, regions where the metric field is straight fare only slightly better and also require many vertex insertions to solve inconsistencies.

Hyperbolic

The "hyperbolic shock" metric field has already been used several times in previous sections (in Sections 4.5.1 and 4.6, for example) and its definition is detailed in Section A.3 in Appendix A. This metric field is characterized by an anisotropy ratio that varies between 1 and 15 and is interesting as its anisotropy ratio does not vary (too much) along the shock, despite the shock being shaped like a sinusoidal curve. We shall refer to the regions of the shock where the eigenvectors change rapidly as the "turns". In these turns, the process of generating a good anisotropic mesh is difficult as the eigenvectors of the metrics are changing rapidly. As larger meshes have already been produced in other sections and very dense refinement was observed at the turns, we here zoom on one of these regions. The result is shown in Figure 4.20.

The size and shape constraints are quickly satisfied and only require around 414 vertices, but the final mesh is composed of 4621 vertices. Indeed, the resolution of inconsistencies is difficult – especially within the shock – and many vertices are required to obtain a consistent mesh despite a relatively low

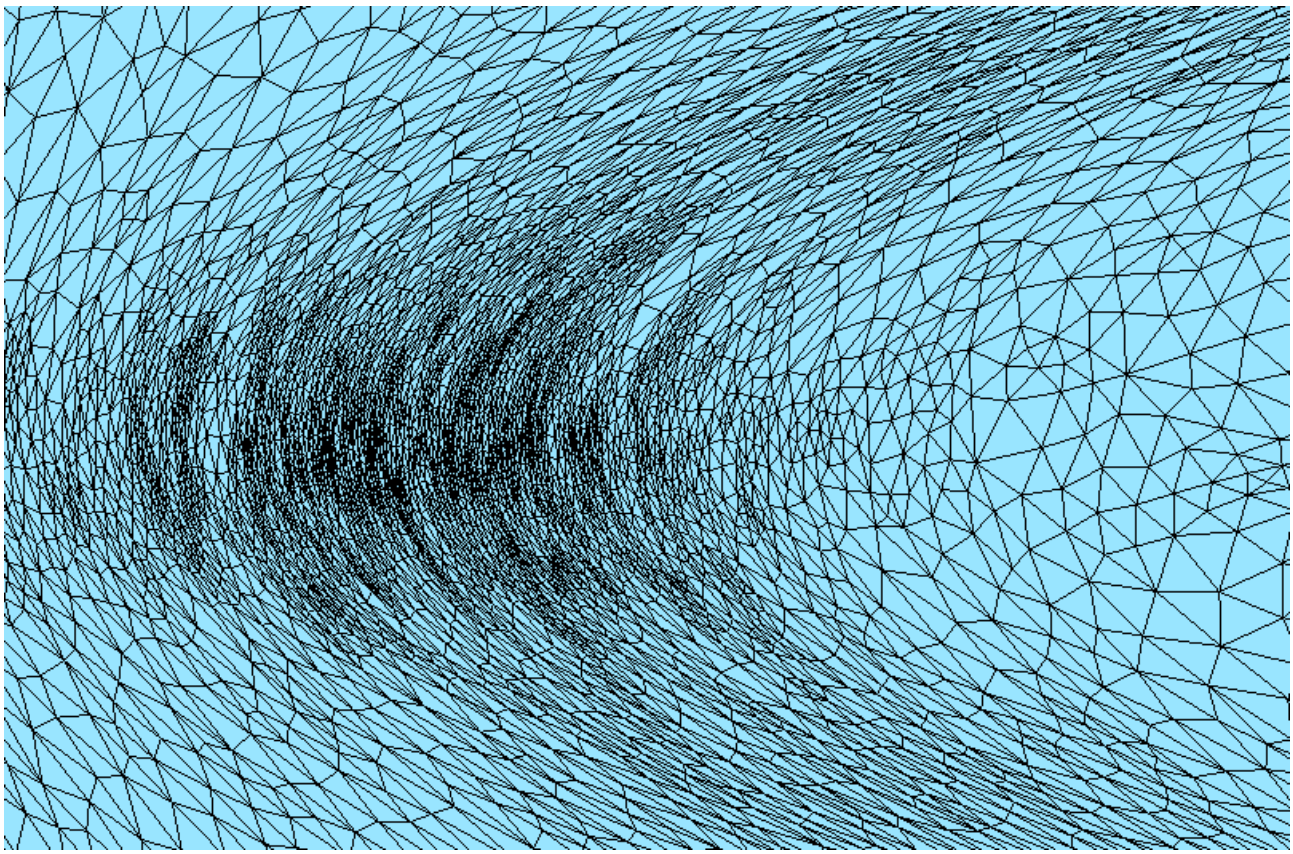

Figure 4.20: Anisotropic triangulation of a rectangle endowed with the hyperbolic shock metric field. The final mesh is composed of 4621 vertices.

maximum anisotropy ratio. Consequently, the metric field is honored (and consequently the sizing field is too), but simplices are often much smaller and then resolution of inconsistencies much longer than what we hoped for.

Swirl

The "Swirl" metric field aims to represent a whirling phenomenon and is detailed in Section A.4 of Appendix A. Contrary to the two previous metric fields, the "Swirl" metric field has a relatively constant anisotropy ratio and is rotating almost everywhere. Figure 4.21 shows the mesh obtained by our implementation for a square of side 6 endowed with this metric field.

The result exhibits the same issues as in the previous experiments: the resolution of inconsistencies is difficult and many vertices are required to solve inconsistencies after all other criteria are satisfied.

4.7.3 Curvature-based metrics fields on surfaces

We now consider the setting of domains embedded in $\mathbb{R}^3$, and the generation of pure surface meshes. The metric field induced by the curvature of the domain is known to prescribe an anisotropy that is – asymptotically – optimal: a mesh whose elements follow this metric field will require the lowest number of element (out of all the meshes) to achieve a given approximation of the domain [57, 107]. It is thus interesting to observe the results produced by our algorithm for this specific metric field.

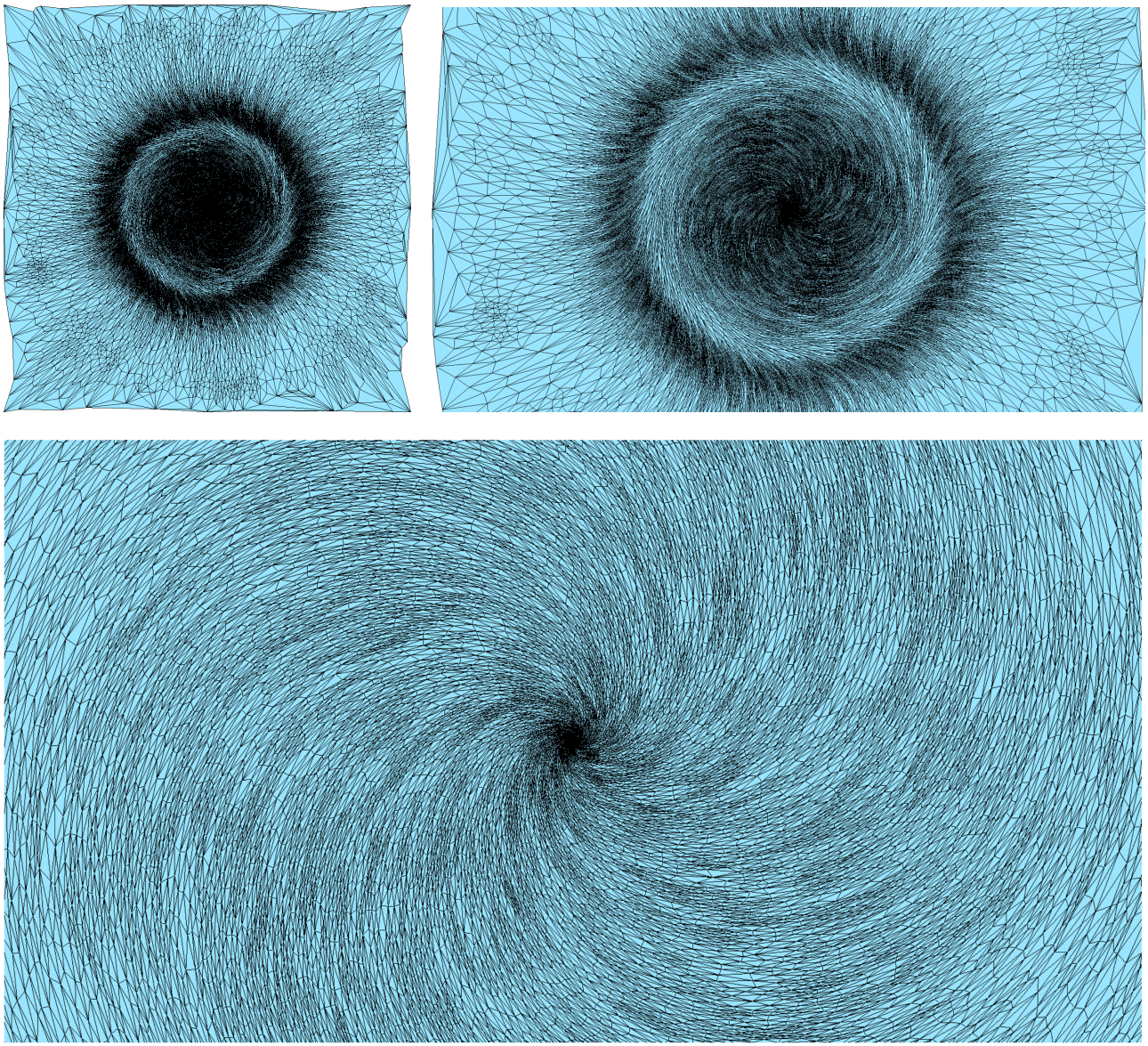

Figure 4.21: A square of side 6 and centered on the origin, endowed with the "Swirl" metric field. The final mesh is composed 28157 vertices and 56342 triangles.

The computation of curvature metric fields for implicit and polyhedral surfaces is defined in Section A.2 in Appendix A.

Implicit surface

An ellipsoid with semi-axes lengths a, b and c (see Section B.6 of Appendix B) is endowed with a curvature based metric field. The ellipsoid is a surface where no curvature vanishes (all points are elliptic), clearing us from the issue that arise from the presence of parabolic points (see Section A.2.1 in Appendix A). We study the variation of the number of vertices when this surface is stretched while keeping the approximation error h_0 constant. The anisotropic mesh is also compared to an isotropic mesh for the same approximation error, which is also generated by our mesher.

To be able to compute the area easily, we consider a specific type of ellipsoid – prolate ellipsoids – which are characterized by $a > b = c$. The surface area of a prolate can be computed exactly, with the following formula:

$$S = 2\pi c^2 \left(1 + \frac{a}{ce} \arcsin(e)\right),$$

where $e^2 = 1 - \frac{c^2}{a^2}$.

The stretching of the ellipsoid is obtained by varying the ratio $r := a/b$. Thanks to the robustness of our implementation, we can handle ratios as high as 10^{17} and produce meshes that conform to such metric field. The approximation error upper bound is fixed at $h_0 = 0.001$ and all the other criteria are set to their default value ($r_0 = 1.0$, $\rho_0 = 3.0$, 60 tries maximum in the `Pick_valid` algorithm, etc.). Figure 4.22 shows two sections of one of the anisotropic meshes.

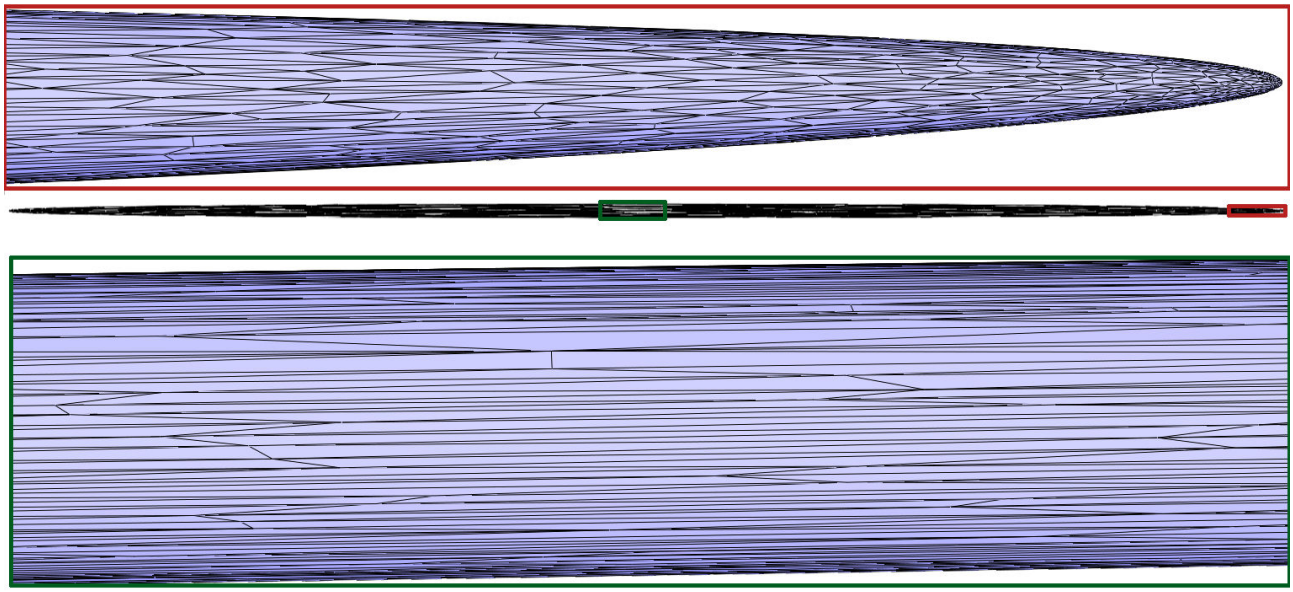

Figure 4.22: Anisotropic Delaunay mesh of an ellipsoid $a = 100$, $b = c = 1$ with $r_0 = 0.1$, $\rho_0 = 3.0$ and $h_0 = 0.001$. The green and red rectangles show a zoom on middle and end sections respectively.

Table 4.4 shows the number of vertices of the final mesh, and the density (vertices per surface unit) for each ellipsoid.

From Table 4.4, we can observe that the size of the anisotropic meshes is barely affected by the stretching: the number of vertices steadily increases as the stretching ratio increases, but this increase is slow. Even for the ellipsoid with a stretching ratio $r = 10^{17}$, fewer than 20000 vertices are required. On the contrary, the density of the isotropic mesh stays relatively constant and consequently the number of vertices greatly increases as the ellipsoid is stretched. One can estimate the number of vertices needed for an isotropic mesh with a ratio of $r = 10^{17}$ from Table 4.4: around 10^{19} vertices would be required to achieve the same approximation error.

Polyhedral domains

The "Oni" and "Cow" domains (see Appendix B) are rough polyhedral surfaces, with only 1435 and 2904 vertices respectively. We endow both domains with the metric field naturally induced by their respective curvatures. Figures 4.23 and 4.24 show the results.

Table 4.4: Isotropic and anisotropic meshes obtained by stretching an ellipsoid with h_0 constant.

	$r = 10$	$r = 50$	$r = 100$	$r = 200$	$r = 500$	$r = 1000$	$r = 10000$
Surface area	99.15	493.58	987.01	1973.95	4934.81	9869.61	98674.23
Number of vertices (isotropic)	24872	126298	250941	502272	1257633	2512782	25133302
Density	250.85	255.88	254.24	254.45	254.85	254.60	254.71
Number of vertices (anisotropic)	6502	6734	6866	6900	7119	7300	7835
Density	65.58	13.64	6.96	3.50	1.44	0.74	0.079

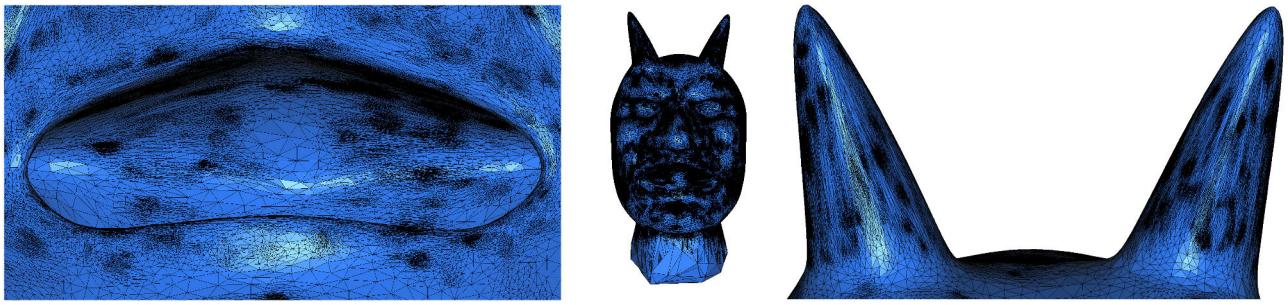

Figure 4.23: The "Oni" polyhedral surface endowed with its curvature metric field, 179214 vertices.

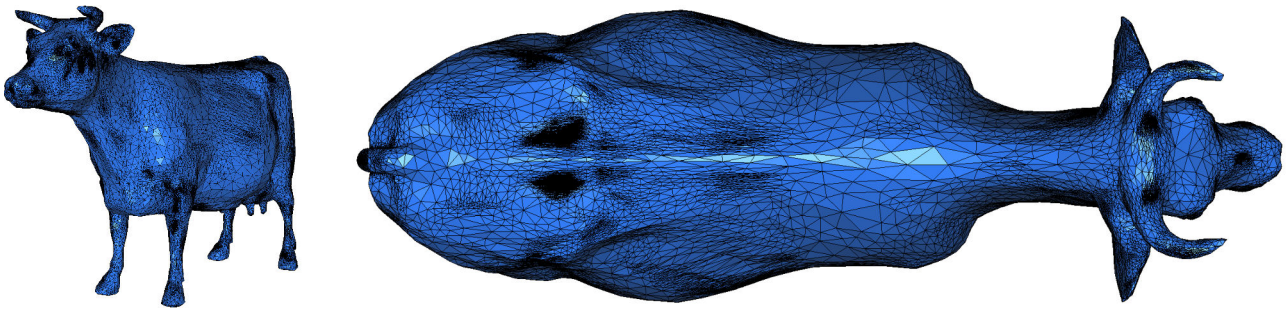

Figure 4.24: The "Cow" polyhedral surface endowed with its curvature metric field, 512514 vertices.

Both results are visually unpleasant, with countless over-refined local regions where consistency is very difficult to achieve.

The many inconsistencies that appear here are caused by the sensitivity of the algorithm to discontinuities and noise in the metric field. Indeed, stars are built using only the metric of the center of the star and any sudden change in the metric field thus requires a great deal of refining to ease the discontinuity. Since these two inputs are sparse compared to the level of detail of the model, the curvature metric field that was computed is rough, which creates these abrupt changes in the two polyhedral domains. While inconsistencies are once again a problem in these results, this experiment mainly illustrates the need to have a relatively "clean" input metric field. Smoothing the metric field will greatly help in the case of rough inputs, as seen in Section 4.8.2.

4.7.4 Shock-based metric fields on surfaces

The polyhedral surfaces "Oni" and "Elephant" are endowed this time with the hyperbolic shock (Section A.3 in Appendix A). As for planar domains, we now evaluate our implementation with shock-based metric fields. The "Elephant" domain is chosen over the cow because its genus is larger and it contains thin cylinder-like parts which are interesting to mesh for Figure 4.25 shows the two results.

Remark 4.7.2 *The extension of two-dimensional metric to a three-dimensional metric is achieved by adding a third (null) coordinate to all vectors and points and defining a third eigenvector orthogonal to the first two with associated eigenvalue* 1*.*

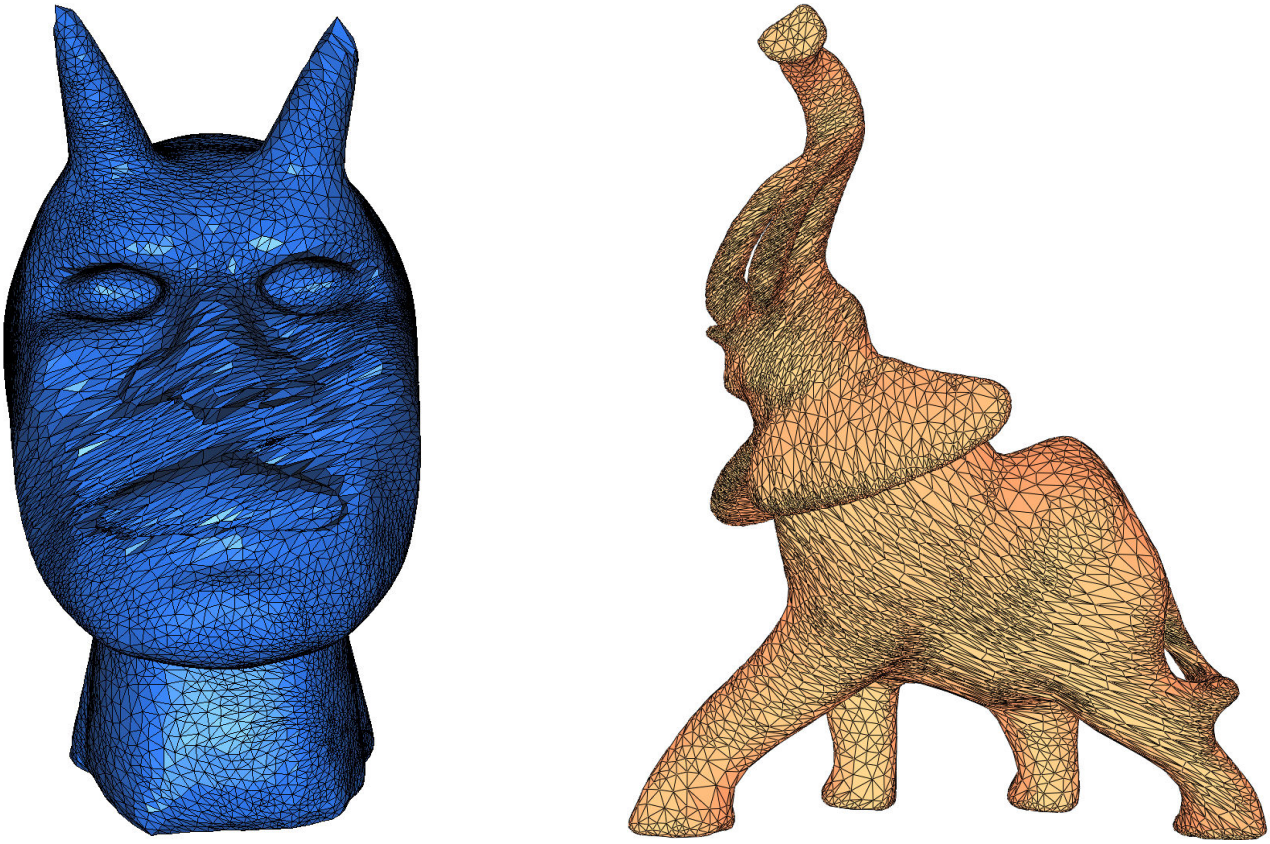

Figure 4.25: The "Oni" (left) and "Elephant" (right) surfaces endowed with the same hyperbolic shock metric field.

Despite a metric field that is not "natural" to the surfaces, the algorithm recovers the topology and achieves the desired approximation, showing the strength of the combination of restricted Delaunay triangulation-based and star-based approaches. As for the number of vertices in the final meshes, the results are similar to those obtained in the setting of a planar domain endowed with a hyperbolic shock: the meshes honor the metric field, but difficulties are encountered in the resolution of inconsistencies, causing a large number of vertices to be required to solve inconsistencies.

4.7.5 Shock-based metric fields in three-dimensional domains

We start our study of the performance of the algorithm of three-dimensional domains with a simple domain, a cube, and a simple metric field, the unidimensional shock metric field. This metric field

contains three wave-like shocks, each along a Cartesian axis (but translated away from the origin), with respective maximal anisotropy ratios 10, 3 and 20. The metric field is detailed in Section A.8 in Appendix B.

This configuration has been used in Section 4.5.1 and the result can be seen in Figure 4.13. Similarly to the other experiments, inconsistencies are troublesome as more than 100000 vertices are needed to obtain a consistent mesh, but fewer than 5000 vertices are required to satisfy all the other criteria.

Radial shock

The classical polyhedral domain "Fandisk" represents a mechanical piece and is characterized by flat faces and sharp features. As explained in Section 4.3.4, these sharp features (edges) are not honored by our implementation as this is not our primary concern. The domain is endowed with a radial shock (see Section A.7 in Appendix A) that can be thought of as the propagation of a wave from a singular point, outwards. We set the maximal anisotropy ratio within the shock to 3 and place the center of the shock on one of the faces of the Fandisk. Figure 4.26 shows the result.

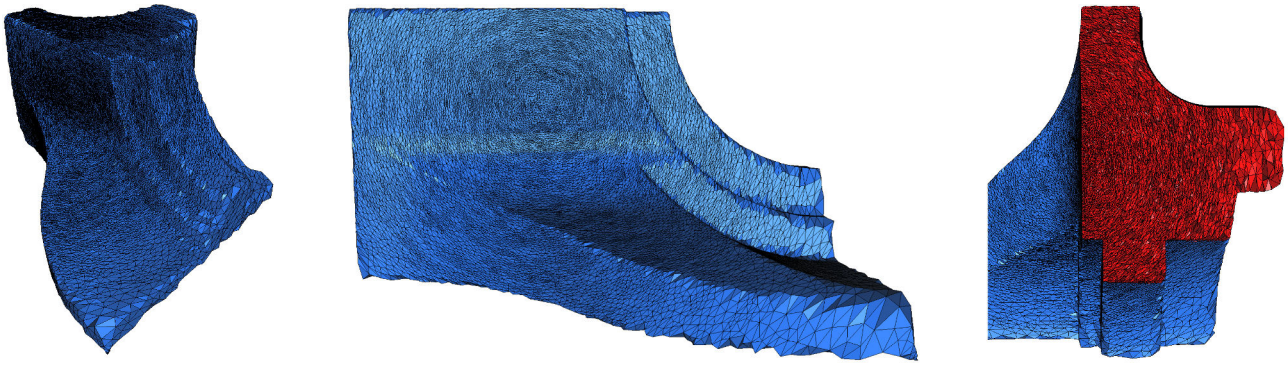

Figure 4.26: The "Fandisk" domain endowed with a radial shock. The mesh is composed of 199283 vertices and 1170393 tetrahedra.

Once again, the sizing, shape and approximation criteria are satisfied very quickly (less than a thousand vertices), but a huge number of vertices is required to obtain consistency (almost 200000), despite a very low maximal anisotropy. If the anisotropy ratio of the shock is doubled (6), achieving consistency requires several millions of vertices.

4.7.6 Mesh quality

We now give detailed statistics for the meshes presented above. The following measures are listed in Table 4.5:

- **The edge length.** To determine how well the algorithm fares at producing a unit mesh, we compute the lengths of edges with respect to the metric at both endpoints. We list the smallest (ℓ_{min}), average (ℓ_{avg}) and largest ℓ_{max} lengths of edges.

- **The distortion.** As the resolution of inconsistencies is the criterion with lowest priority and generates the largest amount of vertices, the final distortion of simplices is a good indication of the distortion at which inconsistencies are finally solved. We compute the distortion of all simplices and list the smallest (ψ_{min}) and average (ψ_{avg}) distortion from all the simplices.

- **The smallest angle.** Angles are a good measure of the quality of a triangulation. For each simplex, we compute its smallest angle (dihedral angle in the case of three-dimensional domains). We list the minimum (α_{min}) and average (α_{avg}) of the smallest angles of all simplices.

- **The quality.** A second measure of quality is computed. For planar and surface domains, we use the formula presented by Zhong et al. [144]:

$$Q = 4\sqrt{3}\,\frac{A}{ph}.$$

where A is the area of the triangle, p the perimeter and h the longest edge (all computed in the metric). For cells, we use the quality estimation introduced by Frey and George [71]:

$$Q = 216\sqrt{3}\,\frac{V^2}{A_\Sigma^3}.$$

where V is the volume of the tetrahedron, and A_Σ the sum of the areas of the four facets (all compute in the metric). Both these measures live between 0 and 1, with 1 signaling the highest quality. We list the lowest (Q_{min}) and average (Q_{avg}) quality over all simplices.

Table 4.5: Statistics on the results shown in the previous sections. The following metric fields are used: Swirl (Sw), Starred (St), Hyperbolic shock (Hy), Curvature (Cu), Smoothed curvature (Sm-Cu) (detailed later), Unidimensional shock (UD) and Radial shock with a maximal anisotropy of 3 (Ra). Definitions can be found in Appendix A.

	ℓ_{min}	ℓ_{avg}	ℓ_{max}	ψ_{min}	ψ_{avg}	α_{min}	α_{avg}	Q_{min}	Q_{avg}
Square (Sw)	6.1×10^{-5}	0.08	1.38	1	1.19	10.92	41.35	0.27	0.76
Square (St)	0.04	0.27	1.8	1	1.20	10.78	26.99	0.31	0.77
Square (Hy)	5×10^{-4}	0.03	0.06	1	1.07	12.56	45.50	0.31	0.80
Fertility (Cu)	0.01	0.50	1.9	1.02	1.82	1.17	39.92	0.03	0.69
Oni (Cu)	1.2×10^{-4}	5×10^{-3}	0.97	1	1.42	0.26	18.78	4×10^{-2}	0.74
Oni (Smooth-Cu)	3.2×10^{-3}	0.04	1.49	1.01	1.39	1.61	41.91	0.05	0.77
Cube 3D (UD)	0.03	0.13	1.01	1	1.08	1.89	43.15	3×10^{-3}	0.73
Oni 3D (Hy)	0.02	0.32	1.96	1	1.08	1.8	42.7	2×10^{-3}	0.73
Fandisk 3D (Ra)	1.8×10^{-5}	0.01	0.07	1	1.03	1.25	40.1	0	0.71

We can make the following observations:

- As expected, the meshes have consistently much shorter edge lengths than prescribed by the sizing information of the metric field due to the over-refinement caused by the inconsistencies.

- The distortion within simplices is small, showing once again that the resolution of inconsistencies is difficult for large distortions, especially in the case of three-dimensional domains. The very low distortion for three-dimensional domains is particularly troublesome as many more vertices are required to achieve a given distortion in 3D compared to planar or surface domains – similarly to the way the number of vertices scales with the square of the sizing field in 2D, but with the cube of the sizing field in 3D.

- The minimum angle in planar domains is consistently much higher than in the other settings. This is a consequence from the circumcenter-based refinement algorithm: it is known that Delaunay-based refinement algorithm guarantees a lower bound on the minimum angle in 2D (up to 20° for Ruppert's algorithm [123]). Since our minimum angle is much lower than the traditional values obtained for isotropic meshes, we can conclude that the `Pick_valid` algorithm has a bad influence on the angles, and thus the quality of simplices.
- Finally, while the average quality is decent in all domains, the element with worst quality has very low quality (close to 0), which is very problematic as a single badly shaped element can be enough to ruin simulations (see Shewchuk [127]). However, no particular care has been given to the optimization of the mesh, which is usually needed to obtain high-quality meshes in the isotropic setting.

Overall, the results are thus relatively disappointing, mainly due to inconsistencies. This phenomenon will be investigated in Section 4.8.

4.7.7 Performance

We investigate in this section the run time of our algorithm on various models. Figure 4.27 shows the elapsed times to generate various meshes in the case of planar and pure surface (only facets are refined) and volume meshes (only cells are refined). Figure 4.28 shows the run time for the generation of a complete mesh (both facet and cell refinement) of the "Oni" domain endowed with the hyperbolic shock.

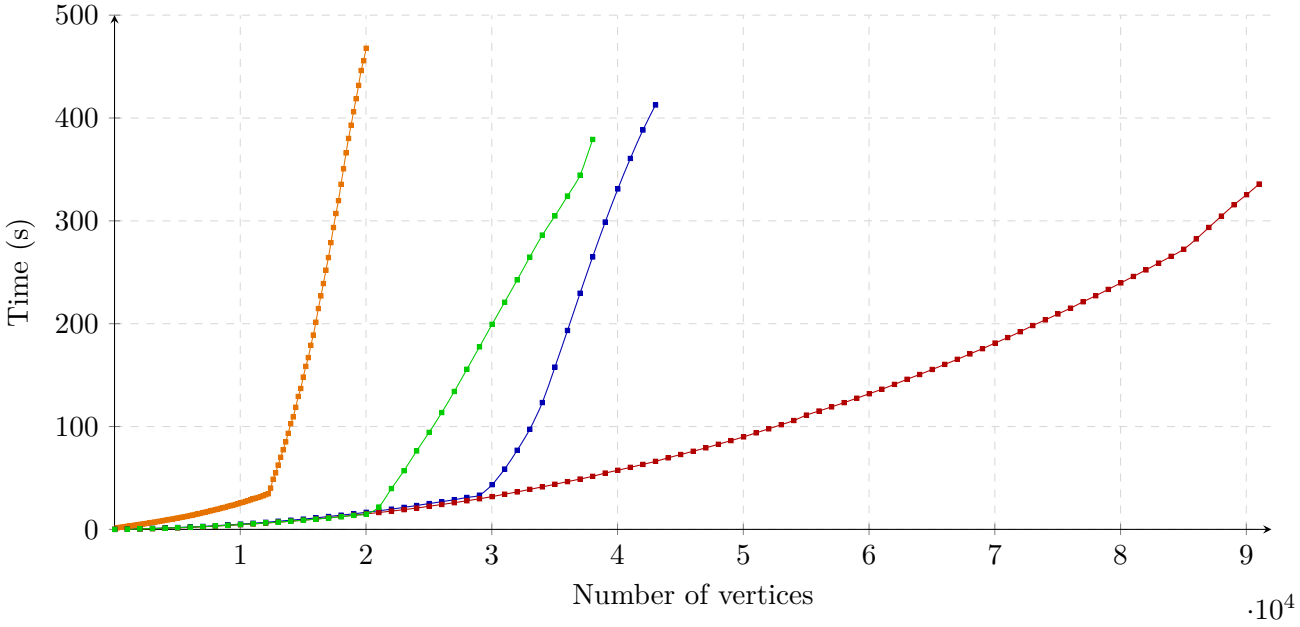

Figure 4.27: Timers for the generation of a planar and surface meshes. The red, blue, and green curves correspond to a square endowed with respectively the starred, hyperbolic, and swirl metric fields. In orange, a pure surface mesh of the Oni surface endowed with its curvature metric field.

While our implementation handles planar, surface and volumes domains, Boissonnat et al. [28] only presented results in the setting of pure surfaces. In this framework, we are approximately ten times

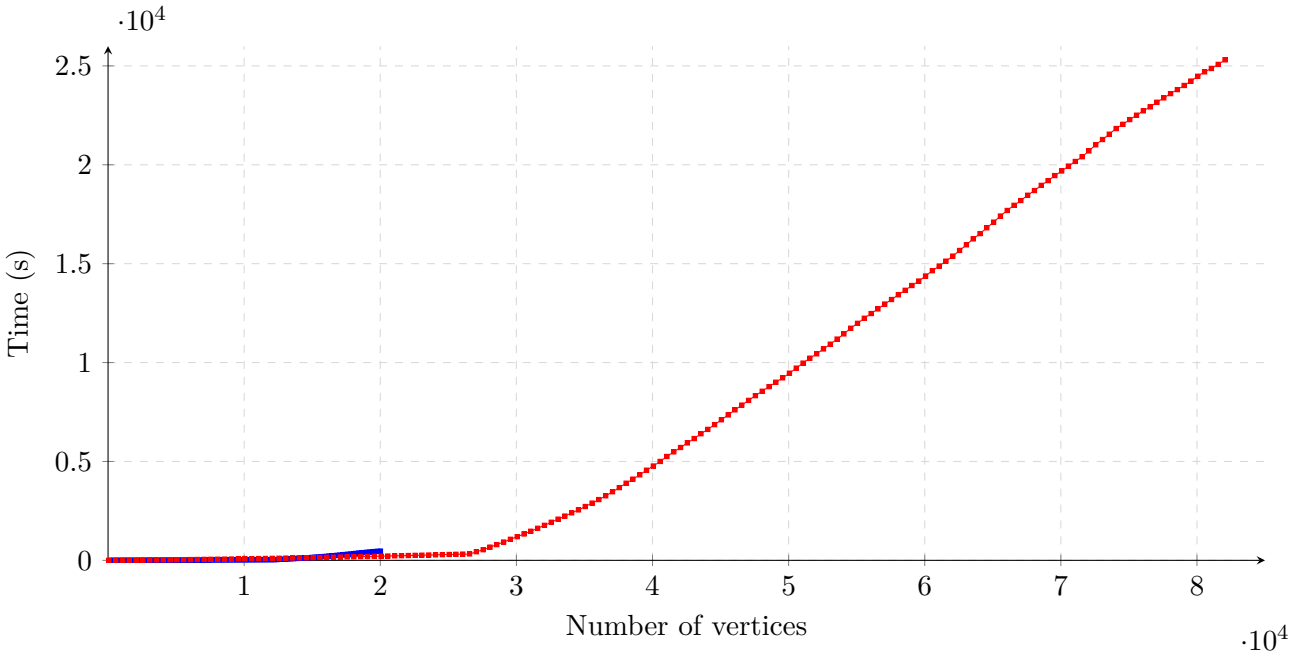

Figure 4.28: Timers for the generation of a surface mesh (blue) and for a three-dimensional domain with boundary (red).

faster than their results (25 seconds to generate the first 10000 vertices compared to 200 seconds). Although this might not seem such a large improvement, our implementation is also significantly more robust as a number of difficult configurations – see Section 4.4 – were not handled. The increased robustness naturally has a non-negligible computational cost.

In Chapter III.6, we shall make use of anisotropic star sets for which we do not need consistency and we will be able to discard some of these robustness tests, further increasing the

Linear speed

The graphs in Figures 4.27 and 4.28 reveal that the computational speed is roughly piecewise-constant: each curve can be broken into two segments. Each segment corresponds to a different types of rules: the rules that insert the circumcenter are cheap and thus the first segment has a low slope; on the other hand, the rules that use the `Pick_valid` procedure are expensive and the slop is much higher.

That the computational speed is much lower when the `Pick_valid` procedure is used is no surprise in light of the previous experiments (see Section 4.6), but the fact that the computational speed is roughly constant for each type of rule might be more surprising. This can however be explained by observing that a cycle of the algorithm (the refinement of a bad simplex) actually has roughly constant complexity. Indeed, despite handling a triangulation at each vertex, these triangulations are kept small and thus the insertion of a point in a star is almost constant time. Additionally, the positions of refinement points ensures that a relatively constant number of stars is modified by the insertion of a new point. The computation of a refinement point is either the evaluation of the circumcenter of the bad simplex or a call to the `Pick_valid` procedure. However, as explained in Section 4.6, finding a valid point is simply simulating the insertion of a random point in the star set and observing whether

it creates bad geometrical configurations. Therefore, as expensive as the procedure might be, it also runs in roughly constant time. Finally, the number of faces to test for badness since the points are well spread.

In a cycle, only two steps do not enjoy constant complexity: the creation of a new star (points that belong in the new stars must be found) and the detection of the stars modified by a new point. However, intensive use of filters greatly reduces the cost of these operations (see Section 4.4.4).

An interesting phenomenon appears in practice: as the number of vertices increases, the computational cost of the filters increases naturally; however, the resolution of inconsistencies becomes easier (as illustrated in Figure 4.17) and the average cost of the `Pick_valid` is thus reduced. While these two phenomena balance each other for a time, the non-linearity created by filter operations can be observed for large meshes – for example on the red curve in Figure 4.27.

Individual cost of functions

Cost-wise, collecting the stars that are in conflict with a candidate or refinement point eclipses all the other procedures: it is by far the most expensive operation in our algorithm. The detection of the stars in conflict with a point p is decomposed in two steps:

- filtering the full star set (using the AABB tree) to obtain a list of stars whose bounding box contains p and have thus a good chance to be in conflict with the point.
- Test if p is actually conflicted with the stars: check for each star if p belongs to a Delaunay ball of a simplex.

Since our refinement algorithm creates a good sampling, the AABB tree returns a roughly constant number of stars potentially in conflict. For similar reasons, there is a relatively constant number of simplices per star. Thus computing whether a star is actually in conflict with the refinement point is a constant approximation. Despite being a constant cost procedure, its cost greatly exceed – in the meshes that we have considered (at most a few millions vertices)– the cost of filtering the stars with the AABB tree, which has non constant cost.

As the percentage of stars in conflict with a point within the set of filtered stars depends on the metric field and the domain as the bounding boxes are axis-aligned. This explains the different computation speeds for different domains and metrics in Figure 4.27.

Comparison with isotropic mesh generators

As our mesher is strongly based on the isotropic mesh generators Mesh_2 and Mesh_3, it is interesting to compare its computational speed with theirs.

We compute the runtime for both algorithms to create meshes of 100000 vertices (the average size of our meshes). The run time of isotropic mesh generators is obtained for very simple domains (a circle and a sphere), with no particular care given to the borders and with parallelism disabled. Mesh_3 takes 91 seconds to generate 96141 vertices, hence a little more than 10000 vertices per second. Mesh_3 (cells only) is slightly slower, with approximately 2000 vertices per second.

When the algorithm destroys simplices by inserting their circumcenter, our implementation has comparatively decent speed and is slightly more than twice slower in 2D and 20 times slower for pure volume meshes. A slower speed is nevertheless expected as each point inserted with our algorithm is the creation of one Delaunay triangulation and the insertion of a point in a roughly constant number of existing triangulations. Additionally, achieving robustness for our algorithm requires combinatorial checks that are not needed when a single triangulation is considered, which naturally takes its toll on the computational speed.

Remark 4.7.3 *The number of stars modified by the insertion of a point naturally depends on the dimension of the mesh. On average (taken over all the results mentioned above),* 6.5 *stars are modified in* $2D$ *and* 14 *stars in* $3D$*.*

These good results must unfortunately be tempered. Indeed, the `Pick_valid` procedure is a time sink and the algorithm is considerably slowed down as soon as the procedure is used. While the small modifications that we have introduced in Section 4.6.1 do help, testing a candidate has a cost equivalent to the insertion of a point in the mesh. In other words, the call to a single `Pick_valid` procedure that uses 60 maximal tries (and fails) will cost approximately 60 insertions of circumcenters. (Note that this was already observed in the experiment of Section 4.6.) Worse, since a great number of vertices are required to solve inconsistencies – a rule that requires the `Pick_valid` procedure – the algorithm is overall very slow. This is especially noticeable in the generation of 3D meshes with boundaries, where the cost of dealing with inconsistencies is extreme.

4.7.8 Conclusion

As proven in theory, the algorithm always terminates and produces a mesh whose anisotropy conforms to the anisotropy prescribed by the metric field. The implementation is robust and reasonably fast when the refinement point is the circumcenter of a bad simplex. However, the presence and resolution of inconsistencies is difficult and requires – consistently over all our experiments – a large number of additional vertices after other criteria are satisfied. This issue is double as the resolution of one inconsistency is much more expensive than the simple insertion of a point.

As explained in Section 2.13 in Chapter 2, a unit mesh with a tolerance of $\sqrt{2}$ is characterized by at most a distortion of 2 between adjacent vertices if the length of an edge is measured in the metric at its endpoints. Inconsistencies generally only can be resolved when neighboring stars have low distortions with experiments showing values of 1.5 for planar surfaces and 1.1 (and lower) for three-dimensional domains (see Section 4.7.6). While it might seem that the difference between a maximum distortion of 1.1 and 2 is not that important, it actually requires a great deal more vertices to achieve. In the context of three-dimensional meshes, this makes it almost impossible to mesh any domain with a metric field whose anisotropy is not completely negligible (say, above 5) without having to insert absurdly large quantities of vertices. For practical purposes, the current state of the algorithm is thus not acceptable.

The next section is devoted to the studies of inconsistencies and of potential ways to remedy this issue.

4.8 Inconsistencies

Experiments in the previous section have showed that inconsistencies prevent the practicality of our meshing algorithm. While it is certain that, given enough time and memory, all the refinement algorithms (be it 2D, surface, or 3D) will provide a criteria-abiding triangulation for any given domain and metric field, the resolution of inconsistencies is extremely costly in term of both time and number of vertices required, usually resulting in overly large meshes.

This has problematic consequences:

- We cannot impose a maximal number of vertices as nothing guarantees that the star set will be consistent with this number of vertices. This makes it impossible to reliably use our refinement algorithm with the purpose of increasing the accuracy of a solution for a constant number of vertices.

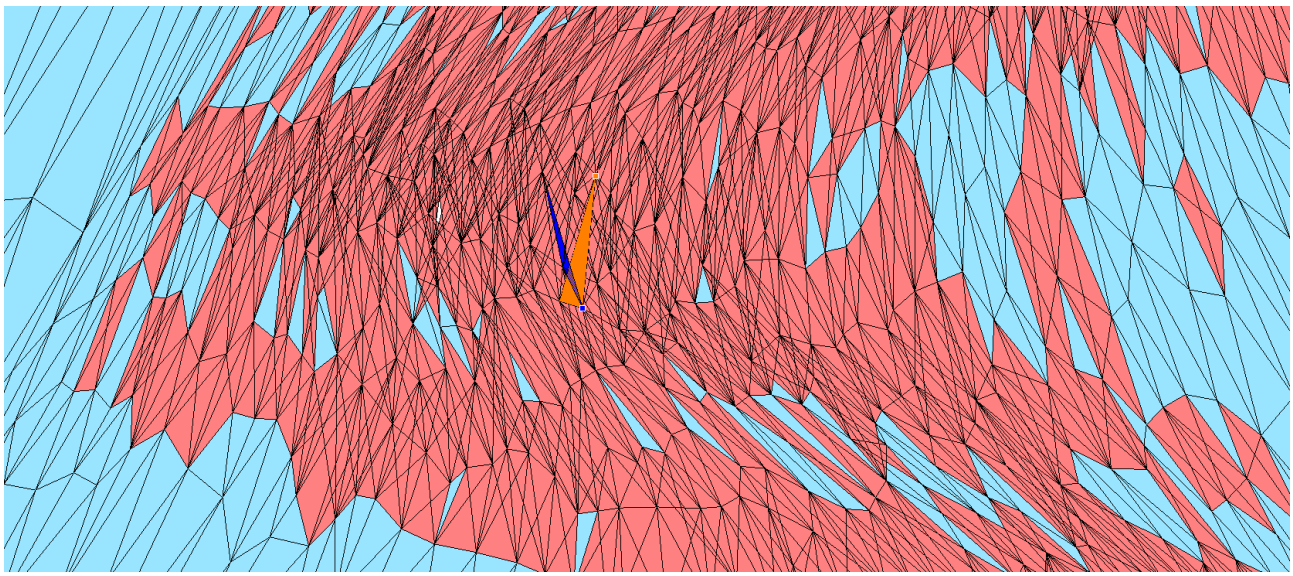

Figure 4.29: An inconsistent region during the refinement process. Inconsistent simplices are drawn in red, consistent simplices are drawn in teal. The blue and orange simplex illustrate one inconsistent configuration.

- We cannot guarantee a rough lower bound on the lengths of edge. In the case of isotropic refinement algorithms, the sizing rule is generally the main criterion that drives the algorithm. Thus, the input sizing field gives an expectation of the length of edges of the final mesh. In our case, each star of a consistent star set will satisfy the size criterion, but the resolution of inconsistencies usually creates meshes whose edges are much smaller than what was expected with the sizing field (see Table 4.5).
- As a consequence of inconsistency issues, the algorithm requires many vertices and calls to the `Pick_valid` procedure and is thus extremely slow.

One might think that the difficult resolution of inconsistency is only due to overly difficult metric fields being considered, but other anisotropic mesh generators, while themselves possessing flaws, have shown to be able to produce close to unit meshes for similar domains and metric fields. For example, the mesh obtained by Liu et al. [73, Figure 1] for a cube endowed with a radial shock metric field with an anisotropy ratio that varies between 1 and 40 (see Section A.6 in Appendix A) is composed of 6554 vertices. For the same domain and metric field, the star set also requires around 6000 vertices for the sizing and shape criteria... and then requires a few millions of additional elements before it becomes consistent. Both cubes are presented in Figure 4.30 with the cube of Liu et al. on the left, and our mesh on the right.

4.8.1 Creation

One might wonder why the inconsistencies are so numerous and difficult to solve. Since the problem of over-refining appears consistently in two-dimensional, surface and three-dimensional domains, it cannot stem from simply the presence of slivery simplices in quasi-cosphericities. Rather, it can be seen as a consequence of both

- the rigidity of the star structure: for two different metrics, the Delaunay balls of a simplex with respect to each metric quickly differs as distortion grows

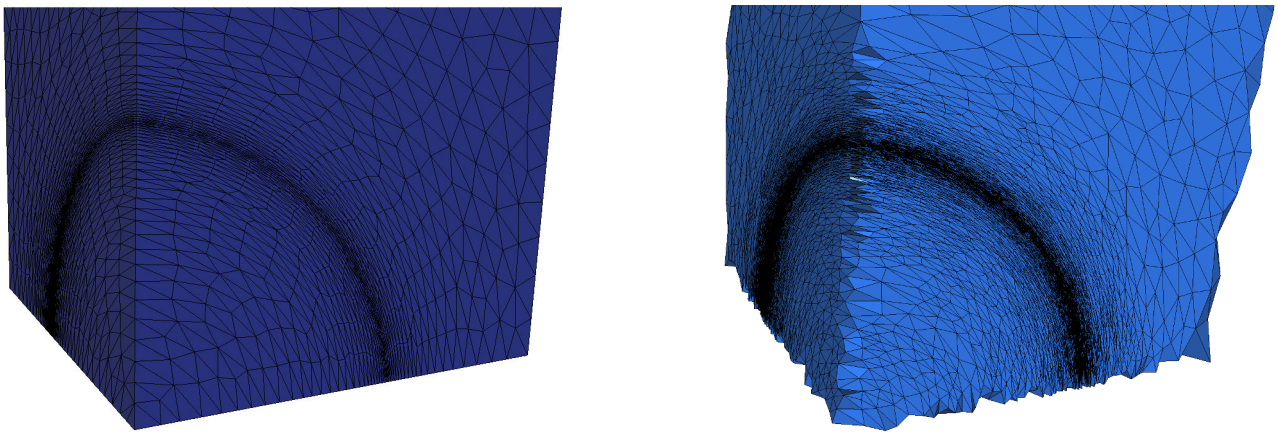

Figure 4.30: 6554 are necessary to produce an almost unit mesh with Liu et al's algorithm. For the same domain and metric field, more than 2 millions vertices are needed for our implementation.

- the isolation of stars: no neighboring metric information is taken into account.

The combination of these two characteristics makes it easy to create inconsistencies, even for relatively low distortions, whose resolution is impossible at distortions that are not close to 1. In comparison, adjacent vertices in the shock region (where the anisotropy is maximal) of the left mesh in Figure 4.30 have distortions up to 5, allowing a much sparser made to obtained.

Remark 4.8.1 *The maximal distortion of* 5 *between two adjacent vertices in the cube on the left in Figure 4.30 implies immediately that this mesh is only a unit mesh with a large tolerance (see Section 2.13 in Chapter 2). Analyzing the length of the edges of their result, we see that indeed the lengths of edges vary between* 0.2 *and* 2.6*; the mesh is thus unit with a tolerance of* 5*, which is thus not completely satisfying either.*

Resolution We aim to find heuristical ways to reduce the cost of consistency, and present the various attempts at achieving this goal. Most of the techniques described are applied (possibly multiple times, and even combined) just before starting the refinement of inconsistent simplices. At this point, we are faced with an inconsistent star set that satisfies all the other criteria (size, shape, sliverity, etc.), which we would like to render consistent without any additional point insertion.

4.8.2 Smoothing

Inconsistencies require low distortion between neighboring points to be solved. Consequently, could the metric field be modified (simplified) to help the algorithm in difficult zones ?

The technique of "smoothing" a metric field, introduced in Section 2.2.3 of Chapter 2, allows for such modifications of the metric field. Different smoothing-based approaches are considered.

- The metric field is preemptively smoothed in input as described in Section 2.2.3 of Chapter 2: each point is reassigned the Log-Euclidean interpolation of himself and his first n (variable parameter) rings on the input domain. In the case of a non-discretized domain (*e.g.* an implicit surface), we first generate a dense-enough isotropic mesh of the domain. This method works very well to smooth out noise: Figure 4.31 shows the same domain as in Section 4.7 but the curvature metric field has been smoothed beforehand. Comparing with Figure 4.23, all the over-refined spots created by abrupt metric changes are gone.

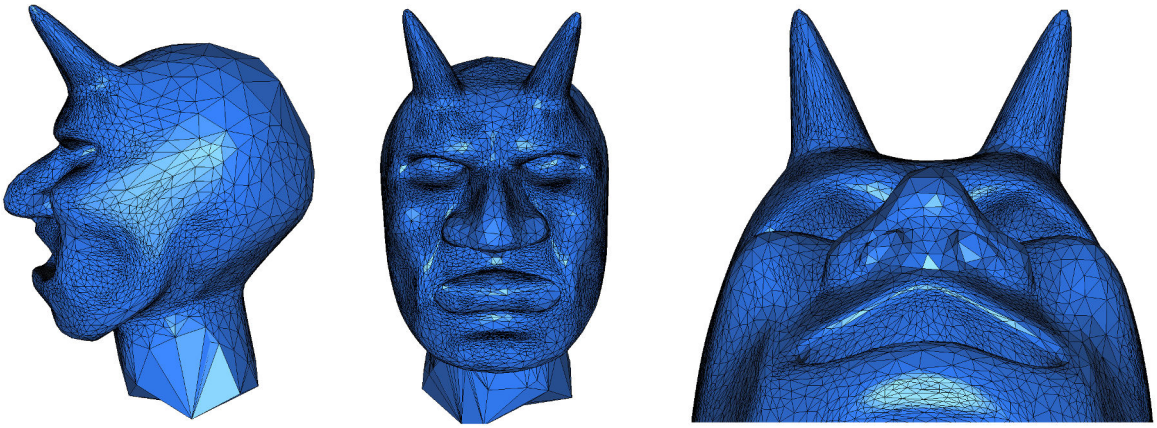

Figure 4.31: The "Oni" domain endowed with a smoothed curvature metric field. The mesh has 8172 vertices and 16097 triangles. Comparatively, 179214 were necessary when the curvature induced metric field was not smoothed (Figure 4.23).

For metric fields that are already smooth (which is the case of most metrics that we have considered), it has however either little influence on the output or massively reduces the anisotropy.

- When the queue of inconsistencies is reached, we smooth the metric field directly on the anisotropic star set: at each point, we attribute a metric defined from a Log-Euclidean interpolation of its k-nearest neighbors (geodesic neighbors in the case of a pure surface mesh). The process is repeated until the distortion between neighbors is below a given bound, consistency is achieved or a maximum number of iterations is reached.

  While we can often achieve consistency without inserting new vertices with this method, it generally comes at the cost of an important loss of anisotropy. Additionally, many smaller issues are present: perturbing the metric of a star after construction is costly and each pass requires a full update of the star set. Finally, it is not assured that the distortion will decrease as we smooth and if it does, the number of passes needed to achieve consistency is not clear and .

- The last smoothing-based approach is slightly different from the previous ones and allows the insertion of new vertices after all criteria (but inconsistencies) are satisfied. However, the refinement point p is attributed a metric G'_p that is not defined as the value of the metric field g at p but as the Log-Euclidean interpolation of the metrics of the stars modified by the insertion of that refinement point. Additionally, the metrics of stars affected by the insertion of p are reciprocally modified slightly by G'_p. With this new definition, the metric field is still continuous and thus the algorithm terminates. Far fewer vertices are required in the case of implicitly-defined metric fields, but the results are plagued with local over-refinement spots (where $g(p)$ and G'_p are too different). This is visually unpleasant, and not much more satisfying than a global over-refinement.

In conclusion, the preemptive smoothing of the metric field is effective to remove noise if the metric field was evaluated numerically, but smoothing does not help with respect to the resolution of inconsistencies. These poor results for inconsistencies are somewhat expected: it is difficult to imagine how an implicitly-defined metric field could be simplified. Consider a simple straight shock metric field, for example $g(p) = \begin{pmatrix} \exp(x) & 0 \\ 0 & 1 \end{pmatrix}$. The distortion between the point $p = (0, 0)$ and $q = (10, 0)$ is

exp(5). If the endpoints of an edge must have a distortion lower than 1.5, we require a certain amount of vertices along (Ox) (that could be computed, but it is besides the point). The only way to simplify this metric field would be to to reduce the "intensity" of the shock, that is to reduce the distortion between p and q.

Reducing the anisotropy to achieve consistency with a reasonable amount of vertices is often easy to do, but generally results in a complete loss of anisotropy. For example, to obtain a consistent star set of the cube endowed with the radial shock metric field (see Figure 4.30), with 6554 vertices (to match the number of vertices in the cube of Liu et al.), the anisotropy ratio of the shock must be reduced from 40 to... $\sqrt{2}$. (This value was found by simply reducing the maximal anisotropy ratio until the consistent star set had the required number of vertices.) This obviously cannot be considered a viable solution.

4.8.3 Relaxed consistency

We study in this section the effect of relaxing the consistency criterion: what if we only require the final star set to be away from a consistent star set by a known set of operations on the star set?

Flip-consistency

Edge flips offer an easy way to switch between different triangulations and have thus been regularly used in mesh generation and optimization. We incorporate the notion of flips in consistency and consider that a simplex σ is *flip-consistent* if σ can be found in all of the stars of its vertices if we allow a simple diagonal flip to happen. Assuming that we manage to create a flip-consistent mesh, we can then apply manual flips to obtain a consistent mesh.

Flip-consistency is, by definition, weaker than consistency and the notion of consistency implies flip-consistency. If a simplex is not flip-consistent, it is said to be *flip-inconsistent*.

We first limit the technique to two-dimensional and pure surface domains, as flip operations in surface stars are simpler to handle than in volume stars. From an implementation point of view, we introduce a flip-inconsistency refinement rule that deals with this relaxed inconsistency: a simplex that is not flip-consistent must be refined. This queue, like the inconsistency queue it replaces, has the lowest priority out of all the queues in the algorithm. The refinement of a flip-inconsistent simplex is similar to an inconsistent simplex: the `Pick_valid` procedure is called and tries to find vertices that would not create flip-inconsistencies.

Results In practice, this approach does not have any significant effect and, consistently across all inputs, many inconsistencies are not flip-consistencies. Figure 4.32 shows the progression of the resolution of inconsistencies in a difficult region of the hyperbolic metric field. While many simplices are flip-consistent, the situation does not improve overall as the algorithm solves inconsistencies.

Allowing more than one flip would certainly help, and even potentially solve inconsistencies. We have chosen however to only allow one diagonal simple flip for two reasons:

- We seek a relatively simple method that extends to all inputs. While flipping in planar domain is relatively straightforward, investigating the combinatorial of stars while allowing multiple flips in 3D seems very difficult.
- If we allow too many flips to happen, the locally uniform anisotropic mesh approach is somewhat lost as we obtain an algorithm that is very close to the work of Mavriplis [102] or Borouchaki [33].

Therefore, this technique is put aside.

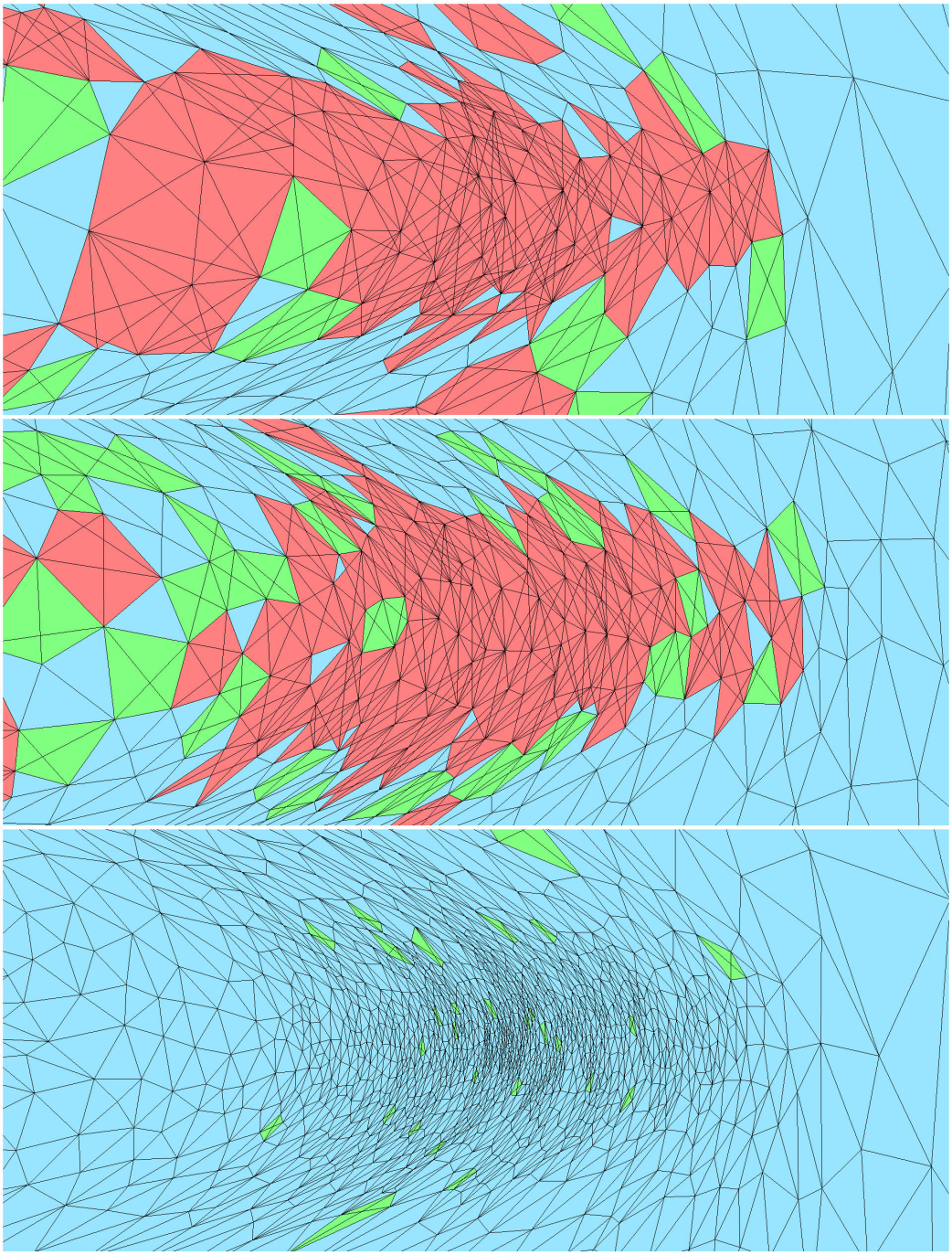

Figure 4.32: Flip-inconsistency and consistency. The consistent facets are in blue, the flip-consistent facets are in green and the inconsistent facets that are not flip-consistent are in red. The domain is the square centered on the origin and of side 1.5, but the images are zoomed in on a difficult part of the hyperbolic metric field. The full star sets possess 1000 (top), 2000 (middle) and 8450 (bottom) vertices. Note that the bottom picture is slightly zoomed out to give a larger view of the difficult region.

Age-based consistency

We define another type of relaxed consistency with the notion of *consistency with stars older than* i: a simplex τ is consistent if τ is present in the stars of its vertices whose index (the position in the star set) is smaller than i (thus were created first and are older). This new type of relaxed consistency will be used in the next section.

4.8.4 Perturbing vertices

The modification of a vertex position is a classic method used in mesh generation, usually to improve the quality of meshes [132].

We attempt to solve inconsistencies without inserting new points by moving points, with the hope of finding better positions that would reduce the number of inconsistencies. The candidate new position of a point is picked in a small sphere centered on the point and of radius 1/3 of the distance to its closest neighbor. The validity of the move is tested as in the `Pick_valid` procedure: no small quasi-cosphericities must be created (see Section 4.2.1). This first approach is largely unsuccessful and only removes a negligible percentage of inconsistencies across all the meshes.

A variation using age-based consistency (defined in Section 4.8.3) is considered, with the idea that points are moved such that they do not create problematic configurations with points that have already been moved. The modified algorithm is as follows:

Algorithm 12 Inconsistency solving: Moving points attempt

```
Define MOVES = 0, MAX_MOVES
for Each star S_p of index id_p and center point p do
    if S_p is consistent with stars older than id_S then
        CONTINUE
    while MOVES < MAX_MOVES do
        Move p
        if p is such that all the modified stars are consistent with stars older than id_S AND
          p does not create inconsistencies in unmodified stars older than S then
            Proceed to next star
        else
            Increase MOVES
    Reset p to its original position          ▷ no improvement found in MAX_MOVES moves
Update all the refinement queues
```

A very simple combination of domain and metric field is considered to investigate the performance of this algorithm. The domain is an ellipsoid with $a = 10$ and $b = c = 1$ endowed with the curvature-induced metric field and the same parameters as in Section 4.7.3. This input has very few issues with consistency with the normal refinement algorithm, and requires very few additional points to finish: the final mesh has 6502 and only 212 vertices are required for consistency. We interrupt the algorithm at 6290 vertices (before the inconsistency queue) and try to resolve inconsistency with point displacement.

This is successful: moving vertices with the age-based consistency criterion successfully gets rid of inconsistencies. However a number of simplices do not honor other criteria (size, approximation, shape) anymore after their vertices have been moved. Refining these facets to satisfy once again the criteria takes around 200 insertions and we thus obtain the same number of vertices as if we had simply refined inconsistencies.

In the case of more complicated metric fields and domains, point perturbation has barely any effect on the consistency, whether using full consistency or age-based consistency. This is not completely unexpected: the `Pick_valid` procedure is in a sense already an optimization of the position of points with respect to the consistency criterion.

Weight-based perturbations

Another type of optimization that is often used in mesh optimization is the weighting of vertices: for example, transforming a Delaunay triangulation into a regular triangulation is one of the way slivers can be removed from isotropic meshes [133, 48].

We adapt this technique to our context and modify the underlying structure of stars to be regular Delaunay triangulation instead of Delaunay triangulations. Thanks to our CGAL implementation, this change is trivial and is simply changing the base class of our `Stretched_Delaunay` class to be a weighted Delaunay triangulation instead of a Delaunay triangulation. We then attempt to weight vertices to remove quasi-cosphericities, but results were poor. This result can be explained by the fact that weighting vertices and perturbing their metric (smoothing) are similar operations and by the fact that only for quasi-cosphericities for a low distortion was weighting able to solve an inconsistency without creating another.

4.8.5 Vertices removal

We investigate in this section the necessity of vertices once consistency has been achieved: could it be possible that a large amount of points are simply needed as "scaffolding" and could be discarded afterwards? Removing a point p from the star set is done in two steps, as shown in Algorithm 13.

Algorithm 13 Removing a vertex from the star set

Remove the point p from all the stars S_q in which it appears.
Remove S_p from the set of stars.

Naturally, we must ensure that any star S_q in which p belonged and has been removed has the correct connectivity and is not missing any vertex (due to star cleaning, for example – see Section 4.4.5).

We first attempt to remove any point from the star set, on the condition that this does not create any inconsistency. Consistently across domains and metric fields, only a few points are removed, proving that all the points are indeed necessary.

Removal with relaxed consistency

Behind relaxed consistency is the hope of obtaining an inconsistent star set that we can make consistent using simple operations, and in particular without adding any new point. Using the flip-consistency notion defined in Section 4.8.3, we try to remove points from consistent meshes without losing the flip-consistency of the star set. The algorithm devised is presented in Algorithm 14. Heavy caching of the stars can be used to make this algorithm fast.

The results of this new algorithm are barely better than when using full consistency: across various domains and metric fields, only a handful of points are removed from the mesh.

In conclusion, we can see that no vertex is superfluous in the final mesh.

Algorithm 14 Inconsistency solving: removing points attempt

for each star S_p in the star set **do**
  Remove the corresponding center vertex p from the stars that contain it
  Remove S_p from the star set
  if Any active star is flip-inconsistent **then** Re-insert S_p in the star set

4.8.6 Constrained stars

In an inconsistent configuration, two stars S_p and S_q disagree on which simplices are Delaunay. What would happen if we decided that one star was "correct", say S_p, and somehow forced S_q to constrain itself such that no inconsistency exists with S_p anymore? The following section describes an heuristic to obtain a triangulation for a given metric field g and a given set of points $\mathcal{P}$. This heuristic is based on rebuilding the star set iteratively, but each star S_p only sees a restricted set of points $\mathcal{P}_p$ that varies for each star; this set is picked such that no inconsistencies will be created with already-rebuilt stars.

The initial point set $\mathcal{P}$ can be (and is, in the examples) computed through the classic Delaunay refinement algorithm used in the star set algorithm. The algorithm then iteratively rebuilds the star set as follows: we start by picking one star S_p. The connectivity of S_p is then *imposed* to others: for any simplex $\tau \in \mathrm{S}_p$ and any $q \in \tau$ with $q \neq p$, we must have $\tau \in \mathrm{S}_q$. Additionally, we must not have any simplex $\tau \in \mathrm{S}_q$ with $p \in \tau$ but such that $p \notin \mathrm{S}_p$. These two conditions ensure that no inconsistency are formed. At each step, we consider a point $q \in \mathcal{P}$, and the star S_q is rebuilt according to the previous conditions. The simplices of S_q are then in turn imposed to other stars.

Implementation wise, imposing the simplices of other stars to a star being rebuilt is done by deriving our `Stretched_Delaunay` class from constrained Delaunay triangulations, which are also available in the CGAL library. As in Section 4.8.4, this is a very simple change, implementation-wise. To ensure that the imposed simplices are not broken by other points, we do not insert (in the star being rebuilt) any vertex that is in conflict with an imposed simplex, that is no point that falls in the Delaunay sphere – in the metric – of an imposed simplex is added to the star.

Although we cannot ensure the termination of this algorithm in all dimensions, the algorithm will terminate since every polygon is decomposable into triangles with only interior edges.

Results Figure 4.33 shows the intermediate steps in the rebuilding of an inconsistent star set. The domain is a square of side 5 endowed with the hyperbolic shock metric field. The final mesh has 7739 vertices; in comparison, the final mesh that had to solve inconsistencies had 43495 vertices.

While this method does provide triangulations, the quality of the final meshes is very low in regions that were inconsistent, for example on the "turns" (the regions where the eigenvectors of the metric are rotating) of the hyperbolic shock metric field. This is especially noticeable when the starting star set is mostly composed of inconsistencies. Figure 4.34 shows an example of such mesh; the domain is almost entirely covered with inconsistencies before we refine for the inconsistency rule. Our algorithm terminates and provides a consistent triangulation, but the simplices hardly conform to the prescribed metric and, even visually, the quality is everywhere very low.

Constraining simplices is the first of the methods that we have investigated that manages to create a consistent star set given a set of points. Unfortunately, the solution cannot be called satisfying.

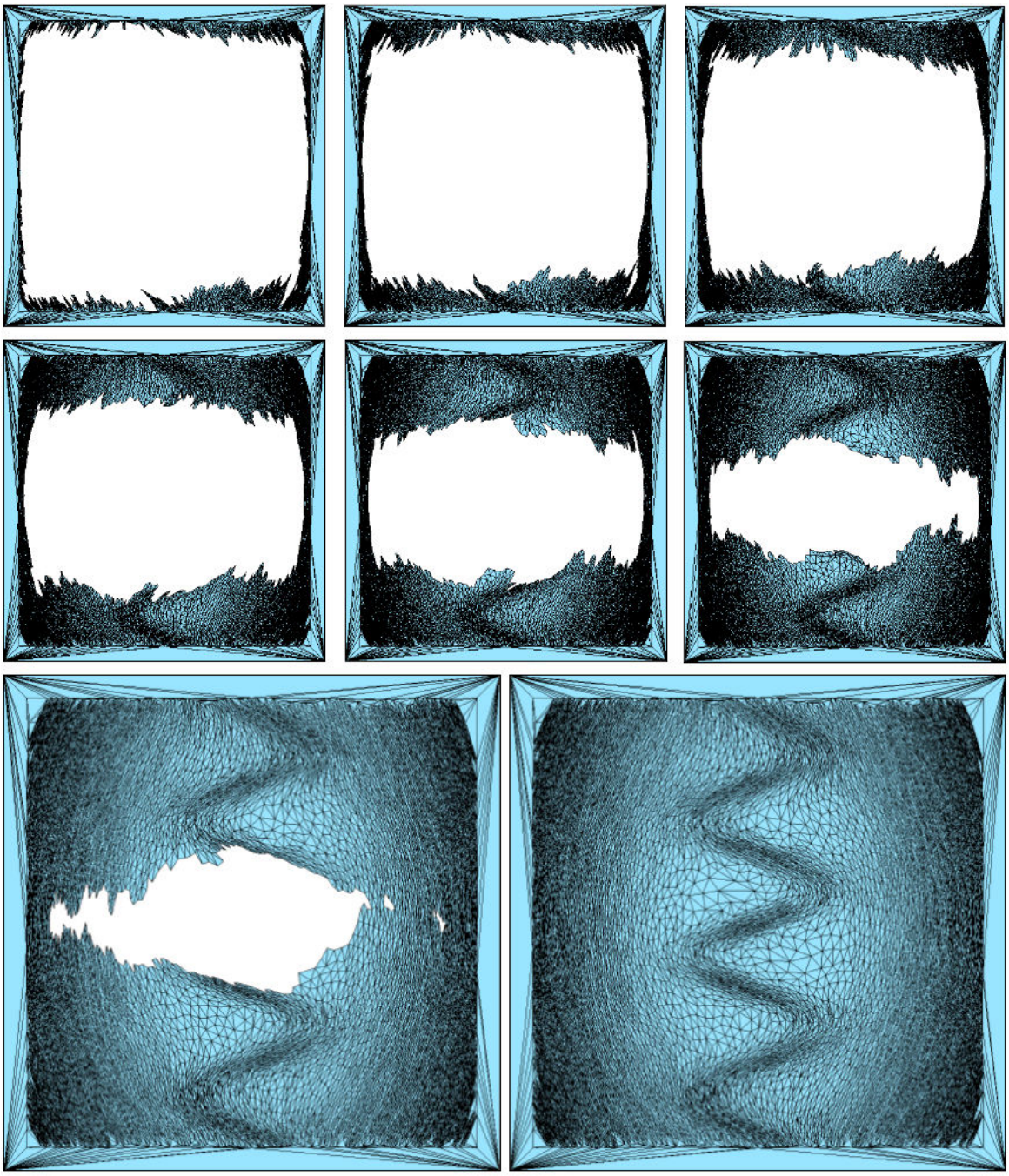

Figure 4.33: Progression of the sweeping $(1000, \ldots, 7000)$ points and final mesh (bottom right).

4.8.7 Using third party vertices

In a last attempt, we considered using points obtained by other Delaunay-based anisotropic mesher generators in the setting of domains and metric fields where a unit mesh was achieved. We extract the position of the vertices from their results, and construct a star set from these points. We generally obtain many inconsistencies. Applying the refinement algorithm to this star set adds a large number of vertices to achieve consistency, as if we had started from scratch.

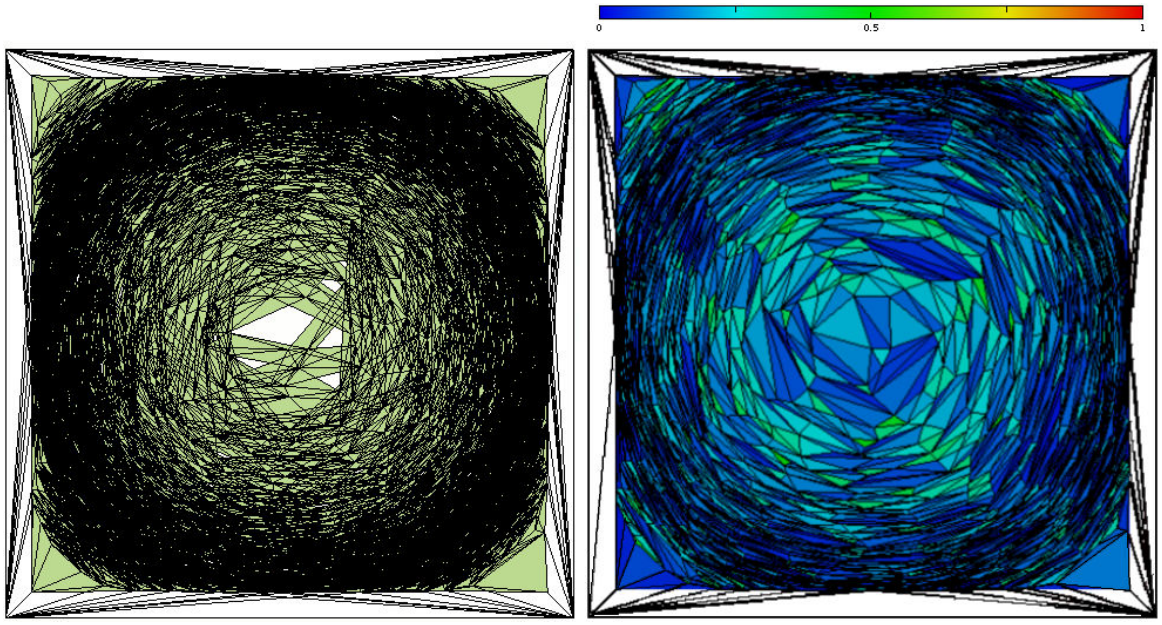

Figure 4.34: Sweeping process with a radial shock metric field. The result has very poor quality.

4.9 Conclusion

We have presented an empirical study of the locally uniform anisotropic meshes framework, introduced by Boissonnat, Worsmer, and Yvinec [30]. During this study, we have introduced minor changes to the theory and provided a robust and generic implementation of the algorithm.

While the algorithm is simple to understand, the computation speed of a naive implementation would be unreasonable. We presented an implementation of the algorithm that is both robust and fast, which is more difficult to achieve. Our implementation is slow compared to other (isotropic or anisotropic) mesh generators, but this is inherent to the algorithm as the insertion of a refinement point must be performed in multiple stars. Additionally, the algorithm relies heavily on the `Pick_valid` procedure to select refinement points, which is extremely costly: testing the validity of a single candidate is more expensive than inserting a vertex in the star set. The large number of invalid candidates generally tested before finding a valid point multiply this cost. While this slowdown can be tolerated in a two-dimensional setting, the additional computational cost becomes important in 3D: inserting a vertex through the `Pick_valid` procedure can have in 3D the same computational cost as the insertion of up to a thousand points in a standard isotropic mesh like the 3-dimensional isotropic mesh generator Mesh_3 of CGAL. The speed of our implementation might though be improved by dissociating our star class from the use of CGAL's classes and using instead a dictionary data structure, as Shewchuk suggests for stars [128], as to avoid the redundancy of each point and simplex usually appearing in multiple stars. However, our CGAL-based implementation allows for a robust and powerful implementation. As an example of this, it was trivial to change stars from being Delaunay triangulations to weighted or constrained Delaunay triangulations in Section 4.8.

The effect of the heavy cost of the `Pick_valid` procedure is amplified by the excessively large number of vertices that must be chosen (using the `Pick_valid` procedure) and inserted to solve inconsistencies, which forms the major drawback of the algorithm. While the algorithm always terminates and provides an anisotropic triangulation that honors the criteria and the metric field, this will almost always require a much larger number of vertices than expected before the star set becomes

consistent. Indeed, the resolution of inconsistencies is generally unsuccessful until very low distortions are achieved between adjacent vertices. Similarly to theoretical bounds on the termination of the algorithm, the theoretical bounds on the quality unfortunately do not translate to good quality meshes either and the lowest and average quality of simplices is generally poor. However, no optimization was performed, which is essential to obtain high quality meshes, even in isotropic settings.

Following these observations, we focused our attention on the resolution of inconsistencies and different approaches were considered to help the resolution of inconsistencies. Unfortunately, no method has been able to significantly improve the situation.

On the bright side, although the algorithm requires low distortion between the vertices of the mesh, it can handle extremely high anisotropy ratios. Additionally, the algorithm is extremely robust to all kinds of input and metric fields, and will provide an anisotropic mesh that conforms to any continuous metric field, given enough time and memory. Finally, despite this shortcoming, the framework of locally uniform anisotropic meshes remains one of the few approaches that offer provable anisotropic mesh generation.

While the approach is not practical to generate anisotropic meshes, we make use of inconsistent anisotropic star sets to help the computation of geodesic Voronoi diagrams. We will then be able to ignore the resolution of inconsistencies, and the refinement algorithm described in this chapter becomes much more useful as it is only a few times slower than an isotropic mesh generator but provides locally Delaunay triangulations whose simplices have good shape and bounded edge size. This use of locally uniform anisotropic meshes will be described in detail in Chapter III.6.

Part II

Anisotropic Voronoi diagrams

5. On the Labelle and Shewchuk approach to anisotropic Voronoi diagrams

The empirical study of Chapter I.4 has revealed that locally uniform anisotropic meshes are not practical. This can be partially attributed to the rigidity of the stars as a Delaunay star does not take into account any metric but the metric of its center point.

We depart from the structure of independent stars and move towards the definition of triangulations as duals of Voronoi diagrams, with the hope that Voronoi cells being dependent on the metric of their neighbors, the increased amount of metric information taken into account will be beneficial to the output triangulation. Several authors have considered Voronoi diagrams using anisotropic distances to obtain triangulations adapted to an anisotropic metric field.

Du and Wang [64] defined the anisotropic Voronoi cell of a point p_i in a domain Ω endowed with a metric field g as

$$\mathrm{V}_g^{DW}(p_i) = \{x \in \Omega \mid d_{g(x)}(p_i, x) \leq d_{g(x)}(p_j, x),\ \forall p_j \in \mathcal{P} \backslash p_i\}.$$

Labelle and Shewchuk [89] had independently the same idea of using only discrete metric information of important points, but their definition of Voronoi cell differs slightly:

$$\mathrm{V}_g^{LS}(p_i) = \{x \in \Omega \mid d_{g(p_i)}(p_i, x) \leq d_{g(p_j)}(p_j, x),\ \forall p_j \in \mathcal{P} \backslash p_i\}.$$

Unfortunately, while the dual of an Euclidean Voronoi diagram is a Delaunay triangulation for any point set (in general position), this property does not hold for a generic distance. Canas and Gortler [37, 38] and Cheng et al. [50] proved that under sufficient sampling conditions, the dual of the Voronoi diagram using the distance of Du and Wang is a triangulation for planar and surface domains, but no result is known for higher-dimensions. In the case of the distance of Labelle and Shewchuk, the associated theory on the embeddability of the dual covers the planar setting, but its extension to higher dimensions faces inherent difficulties. There are thus no proofs in higher dimensions for either diagrams.

We choose to specifically investigate the definition of Labelle and Shewchuk as it fits best with the rest of our work. Additionally, other authors have previously studied this diagram: Canas and Gortler [37] showed that under some sampling conditions, the Voronoi diagram of Labelle and Shewchuk does not possess any orphans, which are parts of a Voronoi cell that are disconnected from the seed. However, they could not prove that the dual of an orphan-free anisotropic Voronoi diagram is a triangulation. Boissonnat et al. [29] proposed an alternative construction of the anisotropic Voronoi diagram of Labelle and Shewchuk as the intersection of a high-dimensional power diagram with a fixed parametrized surface. They proposed a theoretically sound refinement algorithm using this alternative point of view to construct anisotropic Voronoi diagrams whose dual is a triangulation. Unfortunately, as in the original work of Labelle and Shewchuk, this theory is limited to the case of two dimensional domains.

This chapter is divided in two relatively-independent parts. In the first part, we expose conditions on a set of seeds such that the dual of the Voronoi diagram – for the definition of Labelle and Shewchuk – of these seeds is a triangulation, in any dimension, and introduce an algorithm to generate such point

sets. In the second part, we present a refinement algorithm based on the alternate construction introduced by Boissonnat et al. [29]. The refinement algorithm uses a "dual" point of view by constructing a star set, with each star locally approximating the dual of the anisotropic Voronoi diagram of Labelle and Shewchuk. We detail the theoretical issues and investigate its practicality.

Contributions We first give sufficient conditions such that the straight dual (simplices realized as the convex hull of their vertices) of Labelle and Shewchuk's Voronoi diagram is an embedded triangulations in any dimension. Our proof makes use of power protection which can be shown to be linked to the concept of quasi-cosphericities that appeared in the framework of locally uniform anisotropic meshes in Chapter I.4. We present a refinement algorithm to create such point set. These requirements can incidentally be used to prove that multiplicatively-weighted Voronoi diagrams have an embedded dual when generated with similar point sets.

We use the alternative construction of Boissonnat et al. [29] along with an efficient point set reconstruction algorithm, the tangential complex [25] to propose a new refinement algorithm that attempts to build a triangulation that is dual of Labelle and Shewchuk's diagram. The tangential Delaunay complex structure is slightly modified in this setting, which allows for an efficient algorithm despite handling high-dimensional entities. We show that under some density condition, our triangulation is the dual of the anisotropic Voronoi diagram of Labelle and Shewchuk, and can thus be made a triangulation. Finally, we provide an experimental study of the algorithm.

Contents

5.1 Labelle and Shewchuk anisotropic Voronoi diagram

Labelle and Shewchuk [89] introduced an anisotropic Voronoi diagram where each seed sees distances in its own metric. The anisotropic Voronoi cell of a seed p_i is defined by

$$\mathrm{V}_g^{LS}(p_i) = \{x \in \Omega \mid d_{g(p_i)}(p_i, x) \leq d_{g(p_j)}(p_j, x),\ \forall p_j \in \mathcal{P} \backslash p_i\}.$$

We denote $\mathrm{Vor}_g^{LS}(\mathcal{P})$ the corresponding anisotropic Voronoi diagram of a point set $\mathcal{P}$ and with the metric field g. Figure 5.1 shows an example of such a diagram for the hyperbolic shock metric field (see Section A.3 in Appendix A). Although different definitions are possible for an anisotropic Voronoi diagram, we will make almost-exclusive use of this definition and shall therefore refer to the anisotropic Voronoi diagram using the Labelle and Shewchuk distance as simply *anisotropic Voronoi diagram* and will often discard the superscript ^{LS} in the rest of this chapter. When considering any other type of anisotropic Voronoi diagram, we shall explicitly indicate which distance is used.

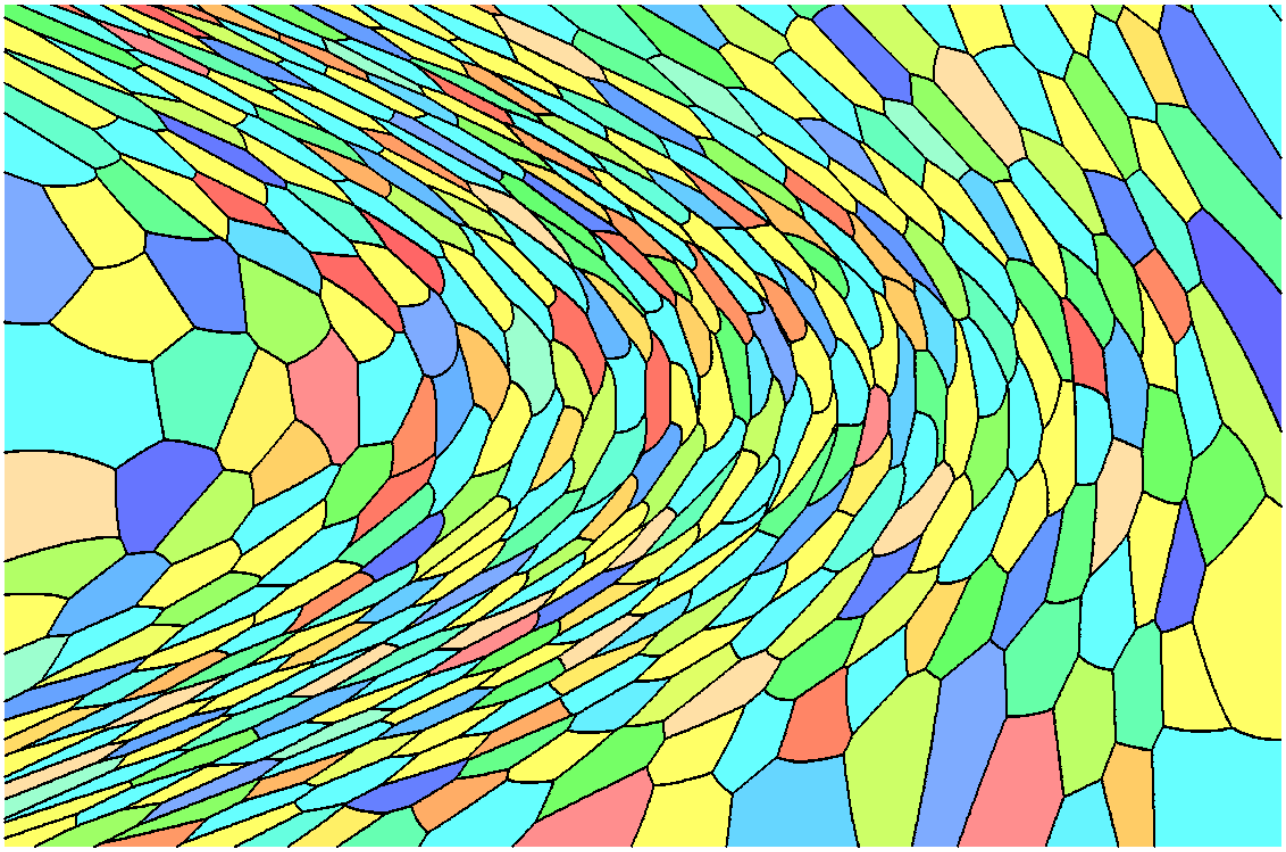

Figure 5.1: Anisotropic Voronoi diagram using the distance of Labelle and Shewchuk.

The anisotropic Voronoi diagram $\mathrm{Vor}_g(\mathcal{P})$ is relatively easy to compute as its bisectors are quadric surfaces (or simply conics in a two-dimensional setting). This can be immediately verified by writing

down the definition of a bisector:

$$\mathrm{BS}(p,q) = \{x \in \Omega \mid d_p(p,x) = d_q(q,x)\},$$

and noting that $d_p(p,x) = d_q(q,x) \iff (x-p_i)^t g(p_i)(x-p_i) = (x-p_j)^t g(p_j)(x-p_j)$.

5.2 Dual of the anisotropic Voronoi diagram

The construction of the dual of a Voronoi diagram must first go through the choice of the geometry of the dual complex: should simplices be drawn using geodesics, segments, piecewise segments? We study here the *straight* dual of the anisotropic Voronoi diagram, meaning the realization of the nerve that uses the classical definition of simplices as the convex hull of a set of points. Since we do not consider any other dual of the anisotropic Voronoi diagram in this chapter, we shall omit the keyword "straight" in the rest of the chapter. The duality between the Voronoi diagram and the Delaunay triangulation is well known in the Euclidean setting, but, as for most other Voronoi diagrams, the dual of the anisotropic Voronoi diagram $\mathrm{Vor}_g(\mathcal{P})$ is generally not a triangulation.

A source of potential issues preventing the embeddability of the dual are *orphans*, region of faces or cells that are disconnected from their seed, which can be present in $\mathrm{Vor}_g(\mathcal{P})$. While orphans are not always responsible for the non-embeddability of the dual of the diagram, they are symptomatic of a poorly-sampled Voronoi diagram (Figure 5.2).

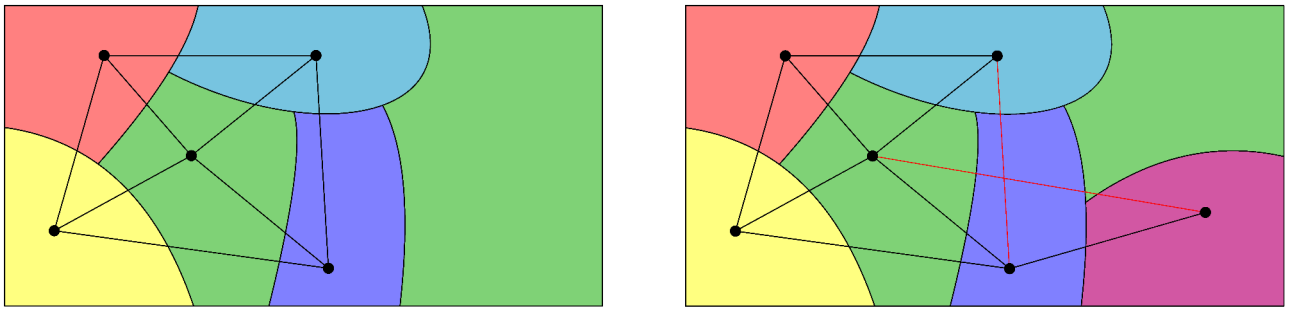

Figure 5.2: Two Voronoi diagrams exhibiting an orphan cell. On the left, the dual is unaffected by the orphan. On the right, the dual is not a triangulation.

In the two-dimensional setting, Labelle and Shewchuk proposed a theoretically sound refinement algorithm that generates a point set for which the dual of the anisotropic Voronoi diagram is a triangulation of the domain. Their proof is based on a visibility argument that ensures that the bisector between two seeds is not too "curved". Unfortunately, the proof cannot be extended to higher dimension – or even surfaces.

We now give sufficient conditions in term of density and quality of the point set for the dual of the anisotropic Voronoi diagram to be an embedded triangulation and introduce a refinement algorithm to point sets that satisfy these requirements. We shall prove the following theorem.

Theorem 5.2.1 *Let $\mathcal{P}$ be a point set over an unbounded n-manifold $\mathcal{M}$ endowed with a metric field g. Denote by $\mathrm{Vor}_g(\mathcal{P})$ If $\mathcal{P}$ is a δ-power protected (ε,μ)-net, with δ sufficiently large and ε sufficiently small, then the dual of $\mathrm{Vor}_g(\mathcal{P})$ using straight simplices is a triangulation of $\mathcal{M}$.*

5.2.1 Detailing of the proof of Theorem 5.2.1

We will introduce in Chapter III.6 the discrete Riemannian Voronoi diagram, a structure that approximates discretely the complete Riemannian Voronoi diagram of a set of points – the Voronoi diagram built using shortest paths on a Riemannian manifold –. Chapters III.7 and III.8 are devoted to the theoretical study of this approach and we expose in particular conditions such that the straight dual of our discrete Riemannian Voronoi diagram is an embedded triangulation, in any dimension.

Due to the many similarities in terms of approach, requirements, and intermediary results between the proofs of the embedding of the dual of the anisotropic Voronoi diagram (using Labelle and Shewchuk's distance) and of the embedding of the dual of our discrete Riemannian Voronoi diagram, we do not detail the proofs in this chapter but simply describe the path that is followed.

Precise computations, lemmas and theorems are however presented in Chapters 3 and III.8 to prove the embedding of the dual of our discrete diagram in the setting of Riemannian Voronoi diagrams, which include the necessary tools and results to prove the embedding of the dual of the anisotropic Voronoi diagram of Labelle and Shewchuk. The results of these chapters – some of them mentioned below – assume arbitrary metric fields, but can be translated to the setting of Labelle and Shewchuk's anisotropic Voronoi diagrams.

5.2.2 Overview of the proof of the embeddability of the dual

We consider in this section an unbounded domain Ω endowed with an arbitrary metric field g. Our approach stems from the observation that as the point set $\mathcal{P}$ becomes denser, the distortion between neighboring vertices decreases and the bisectors become less and less "curved". At a point p, the anisotropic Voronoi diagram $\mathrm{Vor}_g(\mathcal{P})$ is thus well approximated by the uniform anisotropic Voronoi diagram $\mathrm{Vor}_{G_p}(\mathcal{P})$. Using the notion of power protected nets, we can show that the duals are identical and, by combining this information for all vertices, that the dual of $\mathrm{Vor}_g(\mathcal{P})$ is a triangulation.

Orphans

Our approach relies on combinatorial information and is slightly complicated by the presence of orphans, which we thus get rid of preemptively.

Canas and Gortler [38] introduce samples which are samples specifically designed for the anisotropic distance of Labelle and Shewchuk: a point set $\mathcal{P}$ is an *asymmetric* ε-sample if for all $x \in \Omega$, there exists a seed $p \in \mathcal{P}$ such that $d^{LS}_{G_p}(p, x) < \varepsilon$. Using this type of point set, they show that an anisotropic Voronoi diagram can be made orphan-free if ε is small enough.

The notion of nets (see Section 2.12.1 in Chapter 2) that we employ could also be defined asymmetrically. However, we shall use nets defined with respect to g and for the geodesic distance d_g, meaning for example that $\mathcal{P}$ is an ε-sample if for all $x \in \Omega$, there exists $p \in \mathcal{P}$ such that $d_g(p, x) < \varepsilon$. This choice is made for consistency reasons: most of the results that we achieve in other chapters are obtained with nets. It also allows us to define a common refinement algorithm for point sets used in this chapter and in Chapter III.6.

This difference of definition is anyway relatively minor as we can easily construct an asymmetric net from a (Riemannian) net by using the distortion to relate the distances d_g and d_p (see Lemma 3.1.4 in Chapter 3).

Locality

The central idea of our approach is to consider a neighborhood U_p around each seed $p \in \mathcal{P}$ and to approximate over U_p the anisotropic Voronoi diagram $\mathrm{Vor}_g(\mathcal{P})$ by a uniform anisotropic Voronoi

diagram, $\mathrm{Vor}_{G_p}(\mathcal{P})$ with $G_p = g(p)$. The neighborhood U_p is chosen large enough to include both $\mathrm{V}_g(p)$ and $V_{G_p}(p)$.

We then compare the sets of simplices incident to p in the dual of both diagrams – the *stars* $\mathrm{S}_g(p)$ and $\mathrm{S}_{G_p}(p)$ – with the hope of equating them. Indeed, in the setting of a uniform metric field g_0, the Voronoi diagram of a point set $\mathcal{P}$ with respect to g_0 is simply the affine transformation (by the inverse of the square root of g_0) of the Euclidean Voronoi diagram of the point set $\mathcal{P}$ transported to the metric space of g_0. Hence it is clear that the dual of $\mathrm{Vor}_{g_0}(\mathcal{P})$, and specifically the star $\mathrm{S}_{g_0}(p)$ of p, is a triangulation.

Unfortunately, the Voronoi vertices of an arbitrary point set can be arbitrarily close. Consequently, the (obvious) embedding of the dual of an anisotropic Voronoi diagram with respect to a uniform metric field does not guarantee the embedding of the dual of the anisotropic Voronoi diagram with respect to an arbitrary metric field, regardless of its proximity – in terms of distortion – with the uniform metric field.

Separation of Voronoi vertices

The power protection of a set of points $\mathcal{P}$ with respect to a metric g_0 enforces that no point in $\mathcal{P}$ lives close to the Delaunay ball of a simplex in the Delaunay triangulation $\mathrm{Del}_{g_0}(\mathcal{P})$ (see Section 2.12.2 in Chapter 2). Assuming that our seed set is power protected is the key to making our approach based on local approximations viable.

Indeed, we can show that if the point set $\mathcal{P}$ is power protected with respect to G_p, then the Voronoi vertices of the Voronoi diagram $\mathrm{Vor}_{G_p}(\mathcal{P})$ are separated, meaning that the distance between any two adjacent Voronoi vertices is lower bounded. Additionally, assuming that the point set is also a net, we can prove that Voronoi vertices are stable with respect to metric changes: the distance between a Voronoi vertex of $\mathrm{V}_g(p)$ and its combinatorial equivalent in $\mathrm{V}_{G_p}(p)$ can be upper bounded. The combination of these two results, illustrated in Figure 5.3, allows us to deduce that the Voronoi cells $\mathrm{V}_g(p)$ and $\mathrm{V}_{g_0}(p)$ have the same combinatorial information.

Under these conditions, the union of the simplices of the dual of $\mathrm{Vor}_g(\mathcal{P})$ that are incident to p – in other words, the star $\mathrm{S}_g(p)$ – is exactly the star of $\mathrm{S}_{G_p}(p)$. Since $\mathrm{S}_{G_p}(p)$ is a triangulation, $\mathrm{S}_g(p)$ also forms a triangulation.

Combinatorial compatibility of the stars

To obtain a triangulation of the domain, we must verify that the connectivity of all the stars is compatible: a simplex must exist in the star in all of its simplices.

We use the continuous aspect of $\mathrm{Vor}_g(\mathcal{P})$ to show that this is indeed the case. To each k-simplex of the star $\mathrm{S}_{G_p}(p)$ corresponds a $(n-k)$-face F in the Voronoi diagram $\mathrm{Vor}_{G_p}(\mathcal{P})$. By combinatorial equivalence of the Voronoi diagram with respect to G_p and $\mathrm{Vor}_g(\mathcal{P})$, this $(n-k)$-face is also found in $\mathrm{Vor}_g(\mathcal{P})$. Using once again the combinatorial equivalence, the $(n-k)$-face F will appear in the local uniform Voronoi diagram $\mathrm{Vor}_{G_q}(\mathcal{P})$ with respect to all the seeds q such that $F \subset \mathrm{V}_g(q)$ (that is all the vertices of the simplex dual of F).

We have now a consistent star set, which can be proven to be a good triangulation of the domain, similarly to the star set studied in Chapter I.4 (see Section 4.2.4).

5.2.3 Intermediary steps

Our proof can be decomposed in a list of successive results that must be obtained. As explained in Section 5.2.1, all the necessary tools are presented in detail in other chapters for the generic setting

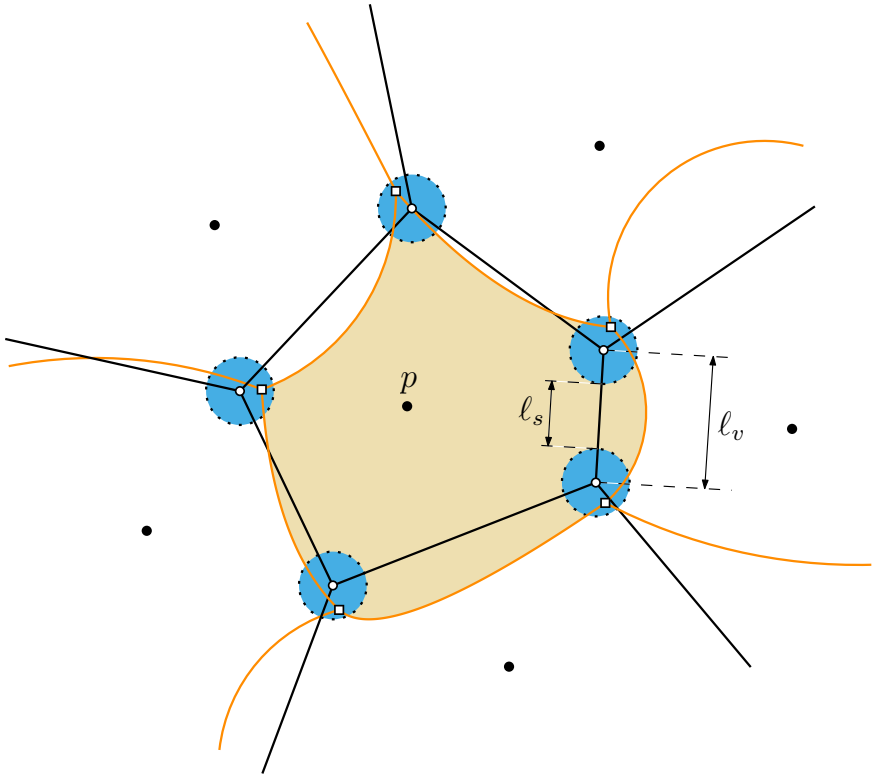

Figure 5.3: The Voronoi diagrams $\mathrm{Vor}_g(\mathcal{P})$ (orange) and $\mathrm{Vor}_{G_p}(\mathcal{P})$ (thick black lines). The blue circles represent the stability regions for the Voronoi vertices. The distance ℓ_v is lower bounded when $\mathcal{P}$ is power protected. When the distortion is sufficiently low, the radius of the blue circles is small and the distance ℓ_s is positive, giving $\mathrm{S}_g(p) = \mathrm{S}_{G_p}(p)$.

of Riemannian Voronoi diagrams. We adjoin links to the proofs.

Remark 5.2.2 *Since a uniform metric field is only a linear transformation away from the Euclidean metric field, we simplify matters in the proofs by assuming that the local metric G_p is the Euclidean metric field $g_{\mathbb{E}}$.*

We must:

- Prove that if the point set $\mathcal{P}$ is a power-protected net with respect to the arbitrary metric field g, and that g and the Euclidean metric field $g_{\mathbb{E}}$ are close, then $\mathcal{P}$ is also a power protected net with respect to $g_{\mathbb{E}}$. This is done in Lemmas 3.4.2 and 3.4.16 in Chapter 3 for the net and power protection properties respectively.

- Prove that for a given set of seeds $\mathcal{P}$ and a seed $p \in \mathcal{P}$, the bisectors of the Voronoi cell $\mathrm{V}_g(p)$ are close to the bisectors of the Euclidean Voronoi cell $\mathrm{V}_{\mathbb{E}}(p)$ when the sampling is dense. This is done from Lemma 3.4.5 up to Lemma 3.4.12 in Chapter 3.

- Prove that in an Euclidean Voronoi diagram built from a power protected net, the distance between two adjacent Voronoi vertices is lower bounded. This is done in Lemma 3.2.4 in Chapter 3.

- Prove that when the sampling is dense, the same Voronoi vertex in the cell $\mathrm{V}_g(p)$ and in the cell $\mathrm{V}_p(p)$ are much closer than their separation distance in $\mathrm{V}_p(p)$. This last part is simply combining

the previous results, and is done in detail in Section 7.3.3 in Chapter III.7 and in Section 8.3.3 in Chapter III.8.

In these conditions, the star $\mathrm{S}_g(p)$ is a triangulation equal to $\mathrm{S}_{G_p}(p)$ and the dual of the anisotropic Voronoi diagram is a triangulation for the reasons detailed in the previous section.

5.3 Protection and quasi-cosphericities

By considering at each seed the connectivity of the diagram using a uniform metric field defined by the metric of the seed, we are in fact bringing the anisotropic Voronoi diagram of Labelle and Shewchuk within the framework of locally uniform anisotropic meshes presented in Chapter I.4, with the small twist that we are here principally looking at the dual of stars (a Voronoi cell and its neighbors).

In the framework of locally uniform anisotropic meshes, the obstacle that prevents a set of stars from being a triangulation of the domain is the presence of inconsistencies in the stars, a configuration where adjacent stars have incompatible connectivities. It can be shown (see Section 4.1.2 in Chapter I.4) that once the distortion between adjacent vertices is small enough, inconsistencies are only created when a point lives outside of the Delaunay G-ball of a simplex σ, but inside the Delaunay G'-ball of σ. Denoting ψ_0 the distortion $\psi(G, G')$, this geometrical configuration is called a ψ_0-quasi-cosphericity (see Figure 5.4, left).

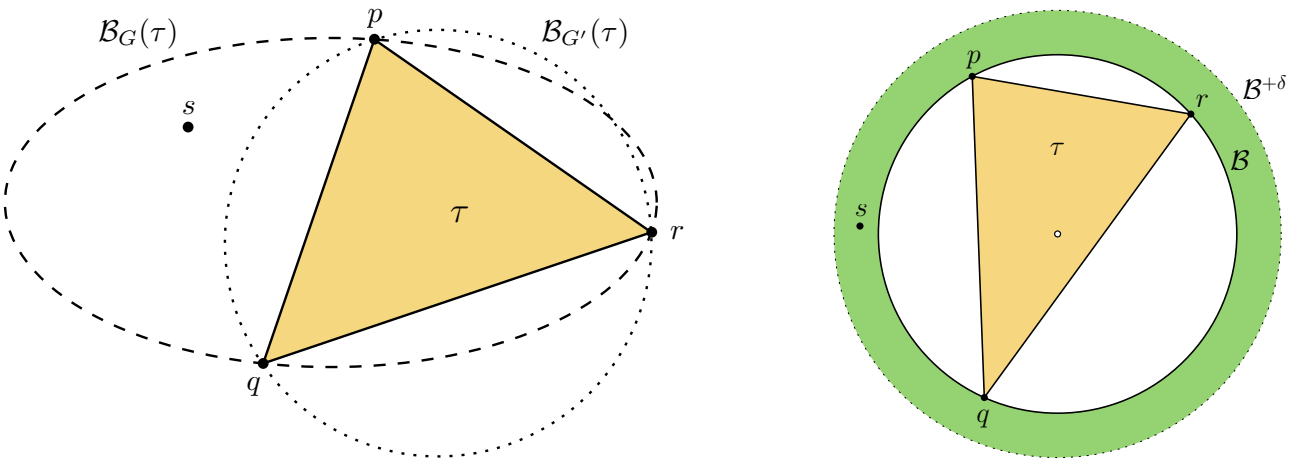

Figure 5.4: A quasi-cosphericity (left) and a simplex τ that is not δ-protected (right).

It may seem surprising that we are able to show that the dual of the anisotropic Voronoi diagram is a star set that can be merged to form a triangulation – and is thus consistent – despite not using the notion of quasi-cosphericities. However, we have used the notion of power protection and, as we shall now explain, power protection and quasi-cosphericities are related.

Power protection and the absence of quasi-cosphericities

The assumption of power protection in fact implies the lack of quasi-cosphericity for a well-chosen distortion. We first give a rough idea of the proof, before detailing the computations.

By definition, if a point set is power protected with respect to G, then there can be no vertex close to the Delaunay G-ball of a simplex of σ. On the other hand, for a small metric perturbation G' of G, the Delaunay G'-ball of σ is close to the Delaunay G-ball. Thus there exists a distortion

bound ψ_0 that is small enough such that all the Delaunay G'-balls are included in the power protected (thickened) Delaunay ball of σ. In other words, there are no ψ_0-quasi-cosphericities.

We now expose the precise computations. Notations used are illustrated in Figure 5.5.

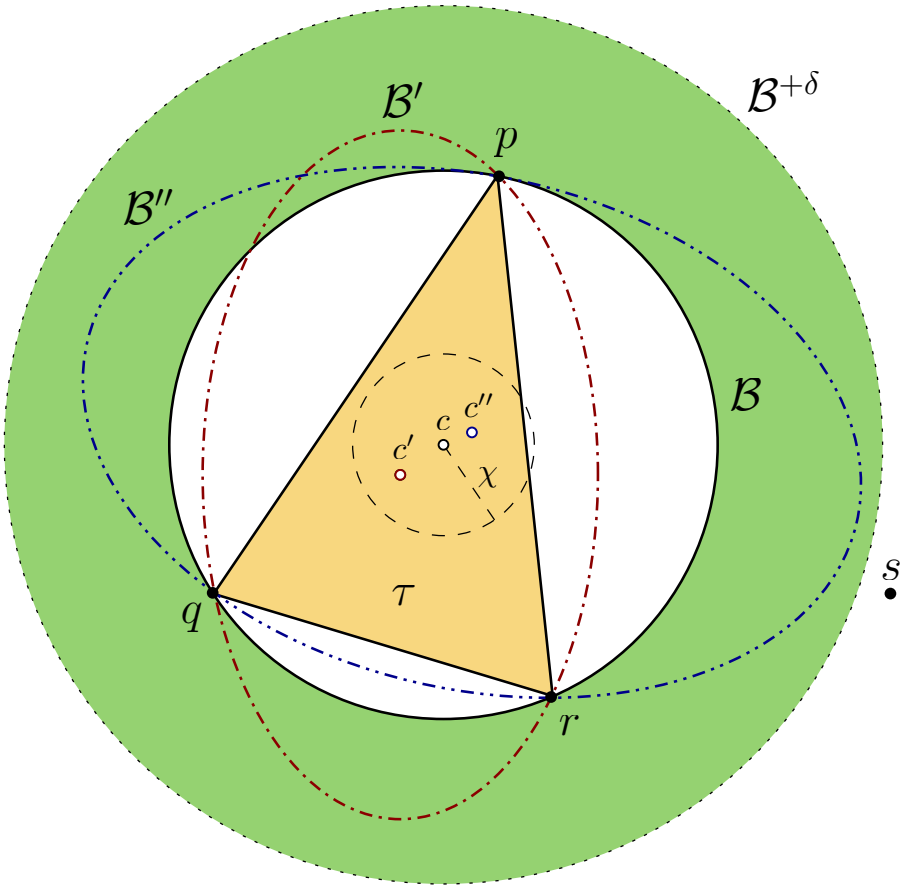

Figure 5.5: The simplex τ is power-protected, which implies no quasi-cosphericities for a distortion bound ψ_0 sufficiently small.

Assume that $\mathcal{P}$ is a δ-power protected (ε, μ)-net with respect to a uniform metric field G. Consider a n-simplex σ, with a dual Voronoi vertex c in $\mathrm{Vor}_G(\mathcal{P})$. By definition of the power protection, we have that for any $p \in \sigma$ and $s \in \mathcal{P} \setminus \mathrm{Vert}(\sigma)$,

$$d_G(c,s)^2 > d_G(c,p)^2 + \delta^2.$$

We wish to find a ψ_0 such that σ is involved in no ψ_0-quasi cosphericities. Consider now a second uniform metric field G' such that $\psi(G,G') < \psi_0$, and denote by c' the dual Voronoi vertex of σ in $\mathrm{Vor}_{G'}(\mathcal{P})$. We want to prove that for any $s \in \mathcal{P} \setminus \mathrm{Vert}(\sigma)$,

$$d_{G'}(c',p) < d_{G'}(c',s), \tag{5.1}$$

that is s is outside of the Delaunay ball $\mathcal{B}_{G'}(\sigma)$, and thus σ is not involved in a ψ_0-quasi-cosphericity.

The most useful property of the distortion is to relate the distances with respect to two metric fields: for x, y in Ω, we have

$$\frac{1}{\psi(G,G')} d_{G'}(x,y) \leq d_G(x,y) \leq \psi(G,G') d_{G'}(x,y).$$

Therefore, if we have $\psi(G,G')d_G(c',p) < \frac{1}{\psi(G,G')}d_G(c',s)$ then Equation 5.1 is satisfied.

We prove in Chapter 3 (Lemma 3.4.14) that Voronoi vertices are stable with respect to metric perturbations: the distance $d_G(c,c')$ is bounded by a function f that depends on the distortion $\psi(G,G')$. (The (ε,μ)-net hypothesis is needed specifically to prove Lemma 3.4.14.) The function f is complicated and tedious to manipulate, but its most important property is that it goes to 0 as the distortion goes to 1. Denote by χ the value $f(\psi(G,G'))$, which is the maximal distance between c and c' for two metrics with distortion $\psi(G,G')$. (Note that this corresponds to the radius of the blue circles in Figure 5.3.) It is clear that

$$\begin{cases} d_G(c',s) > & d_G(c,s) - d_G(c,c') = d_G(c,s) - \chi \\ d_G(c',p) < & d_G(c,p) + d_G(c,c') = d_G(c,p) + \chi. \end{cases}$$

Additionally, we have that $d_G(c,s) > \sqrt{d_G(c,p)^2 + \delta^2}$ by δ-power protection of $\mathcal{P}$ with respect to G. Thus, Equation 5.1 is verified if

$$\psi(G,G')\left(d_G(c,p)+\chi\right) < \frac{1}{\psi(G,G')}\left(\sqrt{d_G(c,p)^2+\delta^2} - \chi\right).$$

Equivalently, it suffices that

$$\left[\psi(G,G')^2\left(d_G(c,p)+\chi\right)+\chi\right]^2 - d_G(c,p)^2 < \delta^2. \tag{5.2}$$

Since χ goes to 0 as ψ goes to 1, it clear that the left hand side goes to 0 (from above) and that there exists a value $\psi_0 > 1$ such that Inequality 5.2 is satisfied.

For this value of ψ_0 we have thus that $d_{G'}(c',p) < d_{G'}(c',s)$ for any metric G' with $\psi(G,G') < \psi_0$ and for any $s \in \mathcal{P}\setminus \mathrm{Vert}(\sigma)$, hence proving that there are no ψ_0-quasi-cosphericities involving σ.

Remark 5.3.1 *At no point have we used that G and G' were metrics (uniform metric fields), and thus this result could be extended trivially to arbitrary metric fields.*

Reverse implication

Power-protection implies an absence of quasi-cosphericities, but the converse is not true. This results from the fact that given a distortion bound ψ_0, the union of all the Delaunay G'-balls with $\psi(G,G') < \psi_0$ does not create a protecting layer around the Delaunay G-ball. Indeed, a Delaunay G'-ball of a simplex σ is an ellipsoid that contains σ and goes through the $n+1$ vertices of σ. Consider the pyramidal double cone spanned by a vertex of σ and its opposite facet. By convexity of a Delaunay ball, no point can live in the region of the cone that does not contain the facet (see Figure 5.6). Thus however small ψ_0 is chosen, we can chose a point as close as we wish to the vertex that spans the cone, without creating a quasi-cosphericity. This implies that there does not exist a $\delta > 0$ such that the simplex is δ-power protected.

5.4 Generation of power protected point nets

The lemmas used in the proof of the embeddability of the dual of the anisotropic Voronoi diagram impose requirements on the sampling and on the power protection of the seed set. We describe in this section how to generate a point set that fulfills these requirements. Many of the Lemmas used in Chapter 2, III.7 and III.8 also require a bound on the distortion within the neighborhood considered.

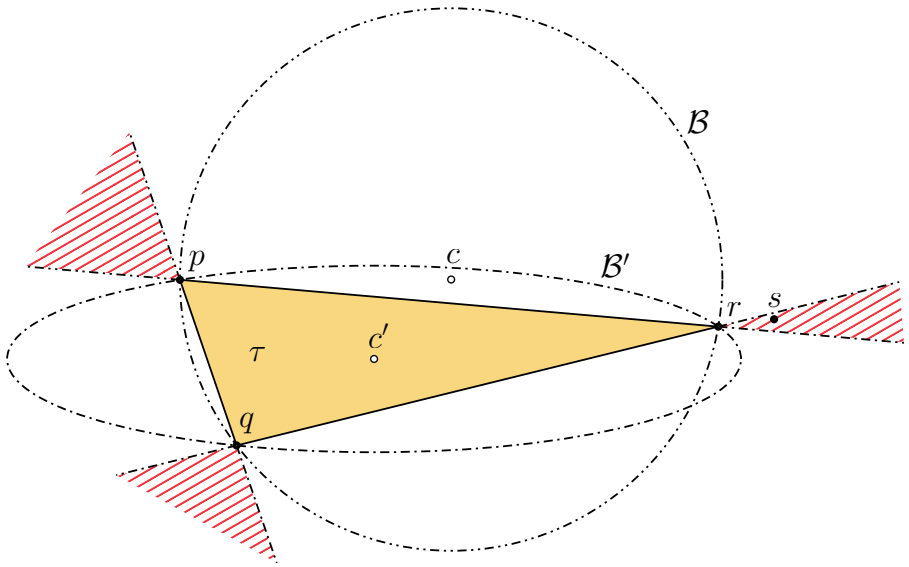

Figure 5.6: Illustration of the fact that an absence of quasi-cosphericities does not imply consistency. The vertex s can be picked in the red dashed zones as closely as wanted to $\mathcal{B}$, without ever creating quasi-cosphericities.

The neighborhood is usually centered on a seed p and is large enough to include the Voronoi cell of p and its neighboring Voronoi cells. Section 3.1 in Chapter 3 describes how to translate a bound expressed through distortion as a bound expressed through sampling, under some mild assumptions on the metric field. We thus wish to generate a δ-power protected (ε, μ)-net that satisfies $\varepsilon < \varepsilon_0$ and $\delta > \delta_0 = \iota\varepsilon$.

Remark 5.4.1 *It is not a mistake that it reads $\delta_0 = \iota\varepsilon$ and not $\delta_0 = \iota\varepsilon_0$: as the mesh gets denser, the power protection required by our intermediary results can be made weaker as the sampling increases. This is very natural: an increased sampling reduces the difference between the Delaunay balls of a simplex, and the δ coefficient can thus be smaller.*

Our algorithm makes use of the similarities between power protection and the lack of quasi-cosphericities that was exposed in the previous section. Similarly to the locally uniform Delaunay method proposed by Boissonnat et al. [30] (see Chapter I.4), our algorithm refines an initial (small) set of points by iteratively inserting vertices whose position is chosen carefully.

In the framework of locally uniform anisotropic meshes, the refinement point of a simplex σ is picked in a small zone around the circumcenter, called the *picking region*, denoted by $\mathrm{P}(\sigma)$. Within a picking region, there might exist *forbidden zones*, formed by the points in $\mathrm{P}(\sigma)$ whose insertion would create quasi-cosphericities with nearby existing points from the point set and must thus be avoided – if possible. In the setting of Boissonnat et al. [30], the thickness of quasi-cosphericities (and thus of forbidden regions) depends on the local distortion and goes to 0 as the distortion becomes small. This allows to prove that for a sufficiently small distortion, the picking region of a simplex is not entirely covered with forbidden zones, and a refinement point that does not create inconsistencies or quasi-cosphericities can be found (Figure 5.7).

As explained in the previous section, the concepts of power protection and quasi-cosphericities are very similar. In the context of a refinement algorithm, avoiding quasi-cosphericities and ensuring power-protection can be achieved with the same idea of inserting vertices away from forbidden zones.

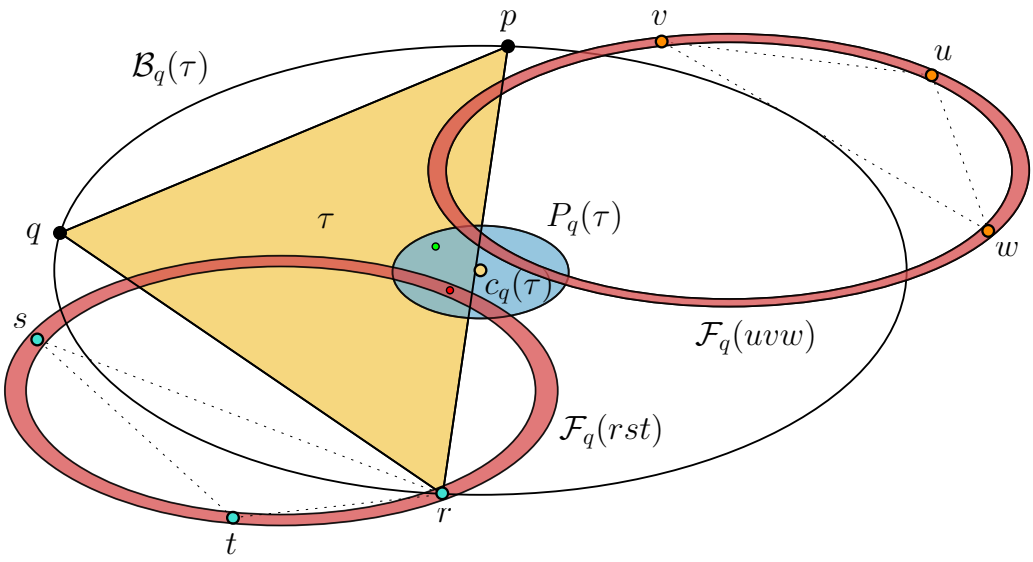

Figure 5.7: A simplex τ, its Delaunay ball $\mathcal{B}_q(\tau)$ (in the metric G_q) and the picking region $\mathrm{P}(\tau)$, in blue. Two forbidden rings (in red) are created by neighboring vertices, one created by s and t along with a vertex of τ, r; and one created by u, v, w. One possible valid refinement point in $\mathrm{P}_q(\tau)$ is drawn in green and one possible invalid point in red.

In our context, the forbidden zones are given by the difference between the power-protected (thickened) Delaunay ball and the Delaunay ball (this corresponds to the rings colored in green in Figures 5.4 and 5.3). The thickness of this new type of forbidden zones is given by the δ_0 coefficient of the power protection.

The key to prove that valid refinements points eventually exist in the framework of locally uniform anisotropic meshes is that the thickness of the forbidden zones goes to 0 as the sampling increases and the distortion goes to 1. Similarly, our bound δ_0 also goes to 0 as the distortion becomes small, which happens as the sampling gets denser.

Remark 5.4.2 *Not only does δ_0 goes to 0 as ε_0 goes to 0, but δ_0/ε_0 goes to 0 as ε_0 goes to 0 (see Equation 3.8 in Chapter 3). This means that our theoretical requirements can use a smaller parameter of the power protection as long as the mesh is denser. This is critical: if δ_0/ε_0 converged to a non-null constant, it would not be possible to prove that there exist a dense enough sampling such that the forbidden zones do not cover the picking region.*

The refinement algorithm that prevents quasi-cosphericities in the framework of locally uniform anisotropic meshes can thus be easily adapted to create power protected nets. The set of seeds $\mathcal{P}$ is initialized with a few random points on the domain and the refinement algorithm is composed of two main criteria:

- A sizing criterion to obtain an ϵ-sample. Simplices are refined with the insertion of their circumcenter.
- A power protection criterion to obtain a δ-power protected point set. A good refinement point is chosen with a procedure similar to the `Pick_valid` routine (see Chapter I.4 for details). If no good refinement point exists (yet) in the picking region, the circumcenter is instead inserted, which lowers the sampling parameter of our point set, thus allowing a lower power protection parameter and increasing the probability of finding a valid refinement point at a further step.

We do not detail the proofs that the algorithm terminates and produces the expected result, as those would be very similar to the work of Boissonnat et al. [31]. A summary of the proof has been given in Section 4.2.4 in Chapter I.4.

The separation of the point set has not been addressed in the description of the algorithm. However, we do not have in theory any hard requirement on the separation parameter μ and a value of μ is implicitly obtained through the lower bound on the insertion radius (which can be computed as in any traditional Delaunay refinement algorithm).

In conclusion, our refinement algorithm allows to obtain a δ-power protected (ϵ, μ)-net with $\varepsilon < \varepsilon_0$ and $\delta > \delta_0$.

5.5 A star-based construction of the dual of the anisotropic Voronoi diagram

In the case of low-dimensional domains embedded in a high-dimensional spaces, the complexity of traditional algorithms to compute Delaunay triangulations or Voronoi diagrams is not satisfying, due to the curse of dimensionality. We introduce a new algorithm to generate good point sets based on the dual of the anisotropic Voronoi diagram of Labelle and Shewchuk. This algorithm adapts the tangential Delaunay complex, a structure that is derived from the Delaunay complex and is used in manifold reconstruction, to the alternative construction of the anisotropic Voronoi diagram as the restriction of a power diagram restricted to a paraboloid proposed by Boissonnat et al. [29]. The complexity of the algorithm is based almost entirely on the intrinsic dimension of the domain, and not on the dimension of the embedding space.

The alternative construction of the anisotropic Voronoi diagram of Labelle and Shewchuk that was proposed by Boissonnat et al. [29] finds its roots in the view of a multiplicatively weighted Voronoi diagram as the intersection of a power diagram and a (fixed) paraboloidal manifold. We recall this equivalence, first detailed by Aurenhammer [11].

5.5.1 Multiplicatively weighted Voronoi diagrams

Multiplicatively weighted Voronoi diagrams are based on the distance

$$d(p_i, x) = \omega_i \left\| x - p_i \right\|.$$

In this case, the bisector of two seeds is a sphere:

$$\mathrm{BS}(p_i, p_j) : \{x \in \Omega,\, \omega_i^2 (x - p_i)^2 = \omega_j^2 (x - p_j)^2\}. \tag{5.3}$$

Reformulating the equality in Equation 5.3, we obtain

$$\omega_i^2 (x - p_i)^2 = \omega_j^2 (x - p_j)^2 \iff x^2 - 2\frac{2\left(\omega_i^2 p_i - \omega_j^2 p_j\right)}{\omega_i - \omega_j} x + \frac{\omega_i^2 p_i^2 - \omega_j^2 p_j^2}{\omega_i - \omega_j} = 0. \tag{5.4}$$

Since $(x - c)^2 = r^2 \iff x^2 - 2cx + (c^2 - r^2) = 0$, Equation 5.4 is the equation of a sphere centered on c and of radius r, where

$$\begin{cases} c &= \frac{2\left(\omega_i^2 p_i - \omega_j^2 p_j\right)}{\omega_i - \omega_j} \\ r &= \left(\frac{2\left(\omega_i^2 p_i - \omega_j^2 p_j\right)}{\omega_i - \omega_j}\right)^2 - \frac{\omega_i^2 p_i^2 - \omega_j^2 p_j^2}{\omega_i - \omega_j} \end{cases}$$

Representing the sphere (c, r) of $\mathbb{R}^n$ as a point in $\mathbb{R}^{n+1}$ (a traditional technique when handling Delaunay triangulations), it can be shown that a multiplicatively weighted Voronoi diagram in $\mathbb{R}^n$ can be obtained as the intersection of a $(n+1)$-dimensional power diagram with the paraboloid $\mathcal{Q} : (x, \|x\|^2)$, where $x \in \mathbb{R}^n$ (see Boissonnat and Yvinec [32]). The seeds of this power diagram are the $\{p'_i = (\omega_i p_i, -\frac{\omega_i}{2})\}$ with the weight $\omega'_i = \omega_i p_i^2$ associated to the point p'_i.

This approach can be extended to the distance $d_g(x, p_i) = (p_i - x)G_i(p_i - x)$ used by Labelle and Shewchuk. The construction is slightly more complicated and relies on a high-dimensional space of dimension $n(n+3)/2$. It is detailed over the next sections, with the final result produced in Theorem 5.5.2.

5.5.2 The power diagram

As for multiplicatively weighted Voronoi diagrams, the anisotropic Voronoi diagram can be obtained as the restriction of a high-dimensional power diagram to a fixed surface. We first detail how the seeds and weights of this power diagram are computed.

Let $\mathcal{M}$ be a smooth m-manifold embedded in $\mathbb{R}^n$ and endowed with a Lipschitz-smooth metric field g. A finite sample of points $\mathcal{P}$ lives on the manifold $\mathcal{M}$. At each point $p \in \mathcal{P}$ is associated a metric $G_p = g(p)$. The metric G_p can be represented by a $n \times n$ matrix: $G_p = (G_p^{l,m})_{(l,m=1,\dots,n)}$. For multiplicatively weighted Voronoi diagrams, the extra dimension can be seen as a way to have a bijection between a point in $\mathbb{R}^n$ and a weight, and a point in $\mathbb{R}^{n+1}$. Here, the metric is a $n \times n$ matrix, with n^2 entries, but by symmetry of G only $n(n+1)/2$ entries are necessary to entirely determine the metric. To each point p are associated:

- the point $q_p = (q_p^{r,s}, 1 \leq r \leq s \leq n) \in \mathbb{R}^{\frac{n(n+1)}{2}}$ with
  - $q_p^{r,r} = -\frac{1}{2}{G_p}^{r,r}$, for $1 \leq r \leq n$ and
  - $q_p^{r,s} = -{G_p}^{r,s}$, for $1 \leq r < s \leq n$.
- The point $\hat{p} = (G_p\, p, q_p) \in \mathbb{R}^{\frac{n(n+3)}{2}}$.
- The weight $\omega_i = \sqrt{\|\hat{p}\|^2 - p^t G_p p}$.

In the following, we use $N = \frac{n(n+3)}{2}$.

Let $\widehat{\mathcal{M}} = \{\hat{x}, x \in \mathcal{M}\}$. We will often need to go from $x \in \mathcal{M}$ to the corresponding $\hat{x} \in \widehat{\mathcal{M}}$, and reversely. For that purpose, we introduce the transformation $h : \mathbb{R}^n \to \mathbb{R}^N$ such that $h(p) = \widehat{p} = (G_p p, q_p)$. The inverse of h is denoted by Π_h.

The point set $\mathcal{P}$ and the transformation h define the point set $\widehat{\mathcal{P}} = \{\widehat{p_i}, p_i \in \mathcal{P}\}$. The high-dimensional power diagram is now built using the weighted point set $\widehat{\mathcal{P}}_\omega = \{(\widehat{p_i}, \omega_i)\,, p_i \in \mathcal{P}\}$. Let $\mathrm{PD}(\widehat{\mathcal{P}}_\omega)$ be the power diagram generated by the $\widehat{\mathcal{P}}_\omega$. The power cell corresponding to the seed $\widehat{p_i}$ ($\widehat{p_i} \in \widehat{\mathcal{P}}$) in $\mathrm{PD}(\widehat{\mathcal{P}})$ is denoted by PC_i.

5.5.3 The paraboloid

In the setting of multiplicatively weighted Voronoi diagrams, the power diagram is restricted to the simple paraboloid $\mathcal{Q} : (x, \|x\|^2)$ (where $x \in \mathbb{R}^n$). The anisotropic diagram is similarly constructed as the intersection of a power diagram and a quadric surface, that we also refer to as "paraboloid". We now introduce this quadric surface.

To each point $x = (x_1, \dots, x_d)$ of $\mathcal{M}$ are associated :

- the point $\bar{x} = (x_r x_s, 1 \leq r \leq s \leq n) \in \mathbb{R}^{\frac{n(n+1)}{2}}$,
- the point $\tilde{x} = (x, \bar{x}) \in \mathbb{R}^{\frac{n(n+3)}{2}}$.

For example, if $n = 2$, $\bar{x} = (x_1^2, x_1 x_2, x_2^2) \in \mathbb{R}^3$, and $\tilde{x} = \left(x_1, x_2, x_1^2, x_1 x_2, x_2^2\right) \in \mathbb{R}^5$.

Let $\mathcal{Q}$ be the n-manifold living in $\mathbb{R}^N$ and defined by the parametrization $\tilde{x} = (x, \bar{x})$. We denote $\{\tilde{x}, x \in \mathcal{M}\}$ by $\widetilde{\mathcal{M}}$. Observe that the manifold $\widetilde{\mathcal{M}}$ is a subset of $\mathcal{Q}$.

As for x and $\hat{x}$, we will often need to switch from x to $\tilde{x}$, and reversely. The transformation $t : \mathbb{R}^n \to \mathbb{R}^N$ is given by $\tilde{x} = t(x) = (x, \bar{x})$. The inverse of t is denoted by Π_t.

Remark 5.5.1 *The notation* Π *used for the inverses of* h *and* t *is not accidental: the transformations* Π_t *and* Π_h *are projections to lower dimensional spaces. These projections are peculiar in the sense that they are also bijections, as it will be shown in Section 5.7.1.*

5.5.4 Equivalence of the constructions

Theorem 5.5.2 proves the equivalence of the constructions.

Theorem 5.5.2 *The anisotropic diagram of* $\mathcal{P}$ *is the image by* Π_t *of the intersection of the power diagram of* $\widehat{\mathcal{P}}_\omega$ *and the manifold* $\mathcal{Q}$*.*

Proof. We have

$$\begin{aligned} d_{p_i}(x, p_i)^2 &= x^t G_i x - 2 p_i^t G_i x + p_i^t G_i p_i \\ &= -2 q_i^t \tilde{x} G_i x + p_i^t G_i p_i \\ &= -2 \widehat{p_i}^t \hat{x} + p_i^t G_i p_i \end{aligned}$$

This implies that $d_{p_i}(p_i, x) < d_{p_j}(x, p_j)$ if, and only if,

$$\|\hat{x} - \widehat{p_i}\|^2 - \left(\|\widehat{p_i}\|^2 - p_i^t G_i p_i\right) < \|\hat{x} - \hat{p_j}\|^2 - \left(\|\hat{p_j}\|^2 - p_j^t G_j p_j\right)$$

□

Figure 5.8 illustrates the different notions with $n = 1$ (and thus $N = 2$). The metric field considered is $g(p) = g(x) = 1 + \cos^2(x)$ and $\widehat{\mathcal{M}}$ is thus parametrized by $(x(1 + \cos^2(x)), -0.5(1 + \cos^2(x)))$.

5.5.5 Refinement algorithm and proofs

Using their alternative construction, Boissonnat et al. [29] proposed a theoretically sound refinement algorithm to obtain a point set for which the Voronoi diagram has a dual triangulation. However, as in the original work of Labelle and Shewchuk, the approach cannot be extended from 2D.

5.6 The tangential complex

We now recall the definition of the tangential Delaunay complex, a subcomplex of the Delaunay complex introduced by Flötotto [70, 23] that we will modify and combine with the alternative construction of the anisotropic Voronoi diagram.

The restricted Delaunay triangulation (see Section 2.11 of Chapter 2), another subcomplex of the Delaunay complex, is an effective tool to accurately capture a domain and its boundary when the

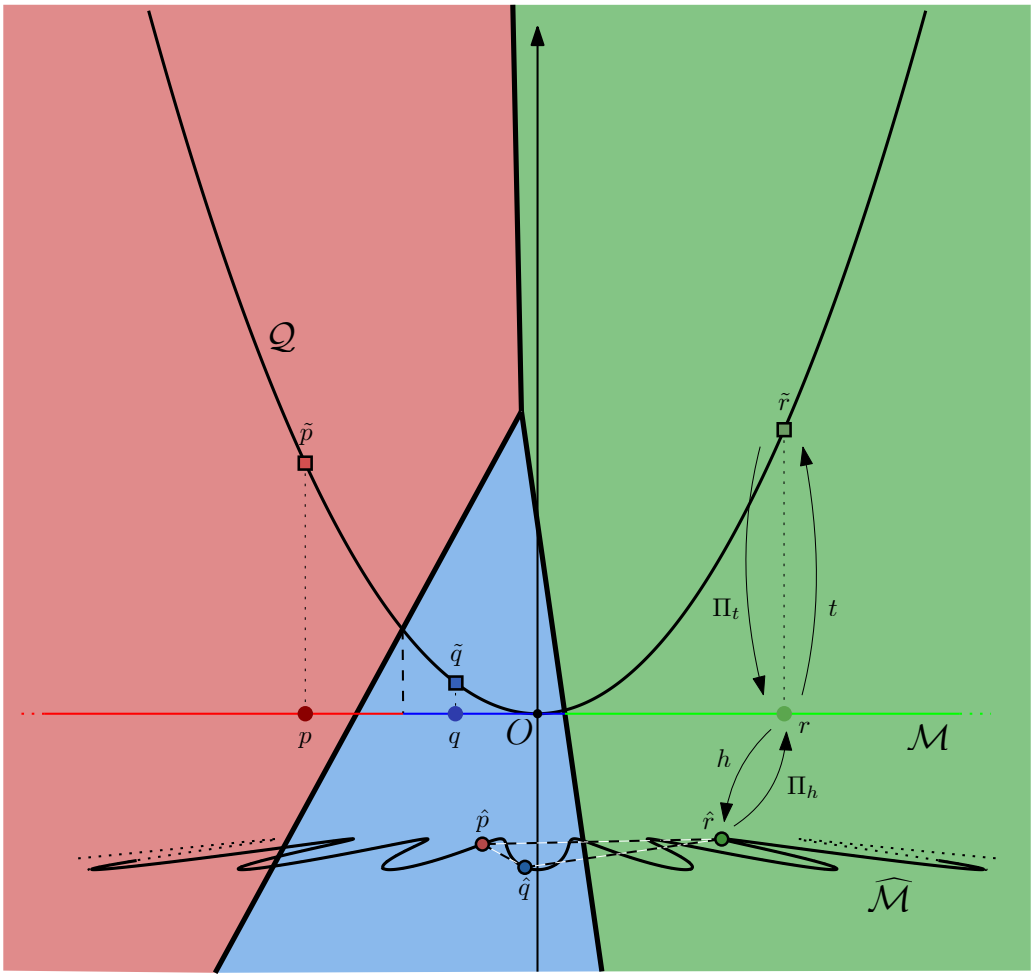

Figure 5.8: Illustration of the notations and concepts ($n = 1$). The manifold is here a straight line, identified with $y = 0$ in the ambient space. The colored areas are the power cells of the power diagram. The black lines are the bisectors of the ambient power diagram. The metric field is $g(x) = 1 + \cos^2(x)$.

domain is known. However, this is not always the case (for example in the context of surface reconstruction). The tangential Delaunay complex aims to approximate the restricted Delaunay complex by considering the restriction not to the domain, but to the tangent spaces of the domain passing through each sample point, which are generally either known or easily computable. The restriction of the ambient Delaunay complex to a tangent space produces a lower-dimensional Delaunay complex, which is locally a good approximation of the domain. Under certain conditions, the local connectivities of all lower-dimensional Delaunay complexes can be merged together to form a provably good triangulation of the domain.

By computing mostly in the tangent spaces of the manifold, the algorithm has the advantage of having a complexity that depends almost exclusively on the intrinsic dimension of the manifold rather than the dimension of the ambient space, making it ideal for the reconstruction of low-dimensional manifolds embedded in high-dimensional spaces.

5.6.1 Star and inconsistencies

In the rest of this chapter, we consider an m-manifold $\mathcal{M}$ embedded in a n-dimensional ambient domain and

The tangential complex approximates at each point $p \in \mathcal{M}$ the restriction to the manifold $\mathcal{M}$ of the exact Delaunay triangulation $\mathrm{Del}(\mathcal{P})$ by the restriction to the tangent plane $T_p\mathcal{M}$. Note that this tangent space is purely geometrical and there is no direct use of Riemannian geometry in this chapter.

The restriction at each sample point $p_i \in \mathcal{P}$ of the ambient n-Delaunay complex $\mathrm{Del}(\mathcal{P})$ to the tangent plane $T_{p_i}\mathcal{M}$, denoted by $\mathrm{Del}_{p_i}(\mathcal{P})$, is formed by the simplices of $\mathrm{Del}(\mathcal{P})$ whose dual intersects $T_{p_i}\mathcal{M}$. If the tangent plane is close to the manifold, the triangulations $\mathrm{Del}_{p_i}(\mathcal{P})$ are – locally – good approximations of the manifold. Consequently, only a local subset of each Delaunay triangulation $\mathrm{Del}_{p_i}(\mathcal{P})$ is kept, the *star*, formed by the set of simplices that are incident to p_i. The star of $p_i \in \mathcal{P}$ is denoted by S_i and the set of stars is denoted by $\mathcal{S}(\mathcal{P})$.

Remark 5.6.1 *From the dual point of view, the subcomplex* $\mathrm{Del}_{p_i}(\mathcal{P})$ *is the dual of the intersection of the ambient Voronoi diagram* $\mathrm{Vor}(\mathcal{P})$ *– the dual of* $\mathrm{Del}(\mathcal{P})$ *– and* $T_{p_i}\mathcal{M}$*. It is known that the intersection of a Voronoi diagram with an* m*-affine space is an* m*-dimensional weighted Voronoi diagram with seeds in the affine space (see Section 2.10). Each restricted triangulation* $\mathrm{Del}_{p_i}(\mathcal{P})$ *is thus the computation of a regular triangulation from a set of (weighted) seeds that are the projections of* $\mathcal{P}$ *onto* $T_{p_i}\mathcal{M}$*.*

As for locally uniform anisotropic meshes (see Chapter I.4), the hope is to merge these stars to obtain a good triangulation of the domain. However, *inconsistencies* between stars might here as well exist: a simplex might not appear in the star of all of its vertices, which prevents the merging of the stars in a triangulation (Figure 5.9).

While introducing the tangential Delaunay complex, Flötotto [70] showed that when the dimension m of the manifold is 1, it was possible to refine an input point set such that merging the stars would create a triangulation of the domain. Boissonnat and Ghosh [25, 24] extended the proof to any dimension m, and proposed a refinement algorithm based on the tangential Delaunay complex that is provably correct and produces a triangulation of a manifold, starting from a rough sampling of the manifold. The triangulation is proven to be isotopic to the manifold and to be close in term of Hausdorff distance. Except for the computation of the refinement point, the complexity of their algorithm only depends on the dimension of the manifold.

5.7 Combined works

The last sections have been devoted to introducing the necessary notions, and we are now ready to combine both approaches. We want to construct a point set $\mathcal{P}$ such that the dual of the anisotropic Voronoi diagram of $\mathcal{P}$ is a triangulation. Summarizing the notions introduced so far, we know that:

- the anisotropic Voronoi diagram can be reformulated as the intersection of a power diagram restricted to a low-dimensional manifold embedded in a high-dimensional space;
- the tangential Delaunay complex and the refinement algorithm of Boissonnat and Ghosh [24] allow to efficiently compute and refine a star set and obtain, in fine, a triangulation that is isotopic to the manifold.

We shall apply the tangential Delaunay complex to the paraboloid $\mathcal{Q}$ defined in Section 5.5.3, with the idea that tangent planes are locally a good approximation of the domain.

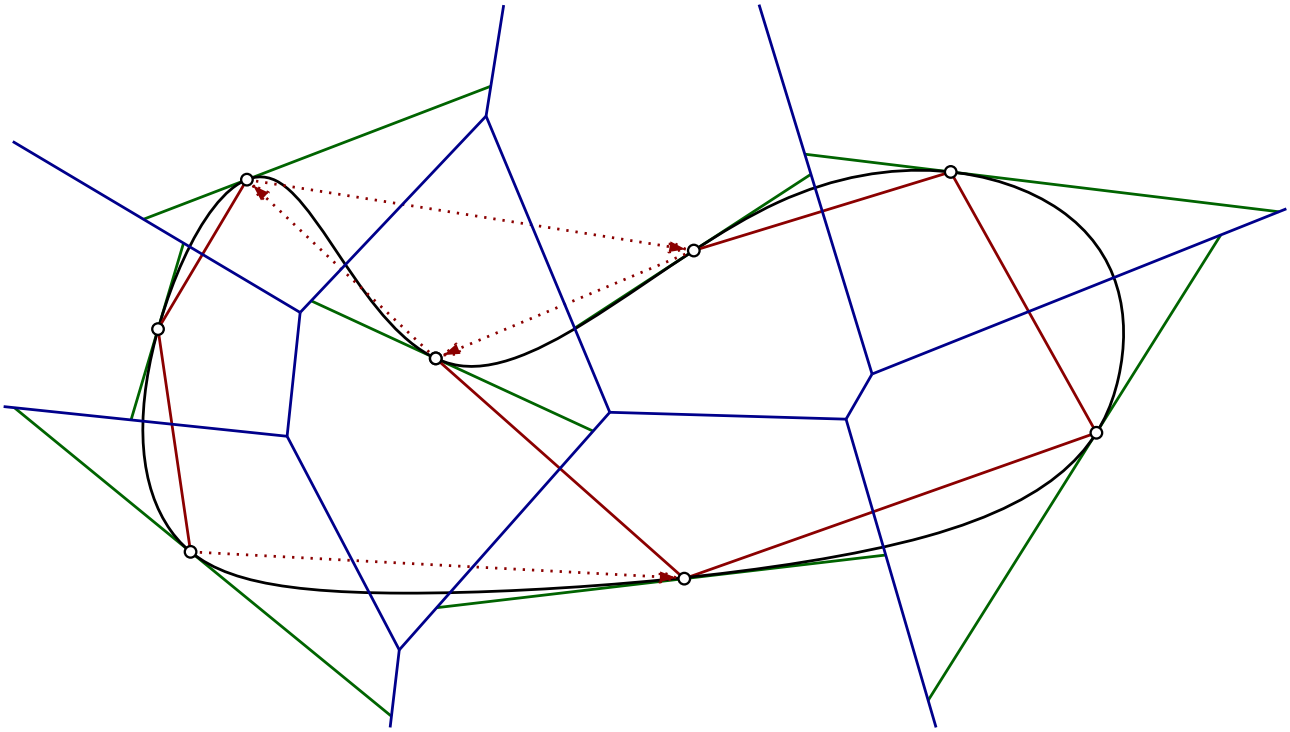

Figure 5.9: An inconsistent tangential complex. Tangent planes are drawn in green, the Voronoi diagram in blue and the stars in red. Inconsistencies are indicated with dotted red arrows.

The two methods cannot be immediately combined for two reasons. Firstly, we do not have an ambient Voronoi diagram, but a power diagram; this is however a minor concern. A more serious departure from previous works is that the basic tangential Delaunay complex assumes that the seeds are on the manifold. Additionally, the tangent planes are taken at the seed of the ambient Voronoi diagram. In our setting, the ambient power diagram is generated by seeds that live on $\widehat{\mathcal{M}}$, but the manifold that we want to construct a triangulation of is $\mathcal{Q}$ and thus the tangent planes are near $\mathcal{Q}$.

In the following sections, we present our modified tangential Delaunay complex. We first introduce its theoretical aspect and the issues that arise when trying to prove its soundness, and then study the practical behavior of the algorithm.

5.7.1 Formulation

We combine both the alternative construction of the anisotropic Voronoi diagram with the tangential complex structure by dissociating the points at which we evaluate the tangent spaces and the seeds of the ambient Voronoi diagram: the tangent spaces are defined as tangent spaces of points chosen on the paraboloid $\mathcal{Q}$ and the ambient diagram will be generated by points on $\widehat{\mathcal{M}}$. We detail now our construction.

The ambient diagram

The ambient diagram whose intersection with $\mathcal{Q}$ we shall approximate with tangent planes, is the power diagram of the seeds $\widehat{\mathcal{P}}_\omega$ (see Section 5.5.2). In the original tangential Delaunay complex, the ambient diagram is an Euclidean Voronoi diagram. Here, we are considering an ambient diagram that is already a power diagram (the seeds are weighted). Construction-wise, this is not an issue as

the intersection of a power diagram with an affine space is simply another power diagram. These computations will be detailed in Section 5.7.1. We denote by $\mathrm{PD}(\widehat{\mathcal{P}})$ the ambient power diagram and by $\mathrm{Del}^{\omega}(\widehat{\mathcal{P}})$ the ambient regular complex. Recall that $\widehat{\mathcal{P}}$ is a set of weighted points.

Center of the star and tangent planes

In the original tangential Delaunay complex, a tangent space at a point p aims to locally approximate the manifold in the neighborhood of the point. Tangent planes are not defined as clearly in our setting since the seed $\hat{p}_i$ do not lie on $\mathcal{Q}$, and we must thus choose meaningful tangent planes to approximate $\mathcal{Q}$. Denote by $T_{\tilde{x}}\mathcal{Q}$ the tangent space of $\mathcal{Q}$ at $\tilde{x} \in \mathcal{Q}$. We now show that considering the tangent plane $T_{\widetilde{p_i}}\mathcal{Q}$ at $\tilde{p}_i$, where p_i is a seed in $\mathcal{P}$, is a good idea.

In the following, we will simply refer to the tangent spaces $T_{\widetilde{p_i}}\mathcal{Q}$ as T_i. The projection on T_i is denoted by Π_{T_i}, or simply Π_i. Recall that for each $p_i \in \mathcal{P}$, we have defined $\widehat{p_i} = h(p_i)$ and $\widetilde{p_i} = t(p_i)$ (see Section 5.5.2).

The restriction of $\mathrm{Del}^{\omega}(\widehat{\mathcal{P}})$ to T_i is denoted by $\mathrm{Del}^{\omega}_{T_i}(\widehat{\mathcal{P}})$, or $\mathrm{Del}^{\omega}_{i}(\widehat{\mathcal{P}})$, and the intersection of the power diagram $\mathrm{PD}(\widehat{\mathcal{P}})$ with T_i is noted $\mathrm{PD}_{T_i}(\widehat{\mathcal{P}})$, or $\mathrm{PD}_i(\widehat{\mathcal{P}})$. We know that the intersection $\mathrm{PD}_i(\widehat{\mathcal{P}})$ can be obtained as a m-dimensional power diagram of a set of seeds living in T_i. These seeds are computed by projecting $\widehat{\mathcal{P}}$ onto T_i, and are denoted by $\widehat{p_j}^i \in \widehat{\mathcal{P}}^i$, with associated weight ω^i_j. The computations of the $\widehat{p_j}^i$ and their weights are detailed in Section 5.7.1. We have thus the alternative notations $\mathrm{Del}^{\omega}(\widehat{\mathcal{P}^i})$ for $\mathrm{Del}^{\omega}_{i}(\widehat{\mathcal{P}})$, and $\mathrm{PD}(\widehat{\mathcal{P}^i})$ for $\mathrm{PD}_i(\widehat{\mathcal{P}})$. Figure 5.10 illustrates the various notations.

Star The star S_i is formed by the simplices that are incident to the vertex $\widetilde{p_i}$, with $\widehat{p_i} \in \widehat{\mathcal{P}}$, in the regular complex $\mathrm{Del}^{\omega}_{i}(\widehat{\mathcal{P}})$. The choice of T_i is reasonable since $\widetilde{p_i}$ belongs to the region $t(\mathrm{V}(p_i))$ of the paraboloid. Indeed, we have $d_{G_i}(p_i, p_i) = 0$ and therefore $t(p_i) = \widetilde{p_i} \in \mathrm{PC}_i \cap \mathcal{Q} = t(\mathrm{V}(p_i))$. In other words, the center of the star will always appear in the star.

To obtain the final mesh of our input point, the connectivity of each star S_i is extracted and applied to the point set $\mathcal{P}$.

Computing the tangent spaces T_i

Contrary to the manifold reconstruction setting in which the tangential Delaunay complex can be used, the manifold $\mathcal{Q}$ is here fully known. This allows the tangent places to be accurately computed, rather than approximated.

A basis $\mathcal{B}$ of the tangent space at $\tilde{x} \in \mathcal{Q}$ can be obtained through the partial derivatives of the parametrization of $\mathcal{Q}$ evaluated at $\tilde{x}$. For example, if $n = 1$, the paraboloid $\mathcal{Q}$ is given by the parametrization (x, x^2). The partial derivative is here a full derivative, giving $(1, 2x)$. A basis of the tangent plane at $\tilde{x} = (0, 0)$ is then $\mathcal{B}_0 = \{(1, 0)\}$.

Computation of the points and weights on the tangent space

Due to the high dimensionality of the ambient space, we cannot afford to compute the regular triangulation $\mathrm{Del}^{\omega}(\widehat{\mathcal{P}})$ nor the power diagram $\mathrm{PD}(\widehat{\mathcal{P}})$. Thankfully, the intersection of $\mathrm{PD}(\widehat{\mathcal{P}})$ with the tangent space T_i is simply another power diagram, $\mathrm{PD}_i(\widehat{\mathcal{P}})$, whose seeds are given by the orthogonal projections of the seeds of $\mathrm{PD}(\mathcal{P})$ on T_i. Indeed, we have:

$$\forall x \in T_i, \|x - \widehat{p_j}\|^2 = \|x - \Pi_i(\widehat{p_j})\|^2 + \|\Pi_i(\widehat{p_j}) - \widehat{p_j}\|^2 .$$

The power cell of p_j in the ambient space is the set of points x such that

$$\|x - \Pi_i(\widehat{p_j})\|^2 + \|\Pi_i(\widehat{p_j}) - \widehat{p_j}\|^2 - \omega_{\widehat{p_j}} \leq \|x - \Pi_i(\widehat{p_k})\|^2 + \|\Pi_i(\widehat{p_k}) - \widehat{p_k}\|^2 - \omega_{\widehat{p_k}},$$

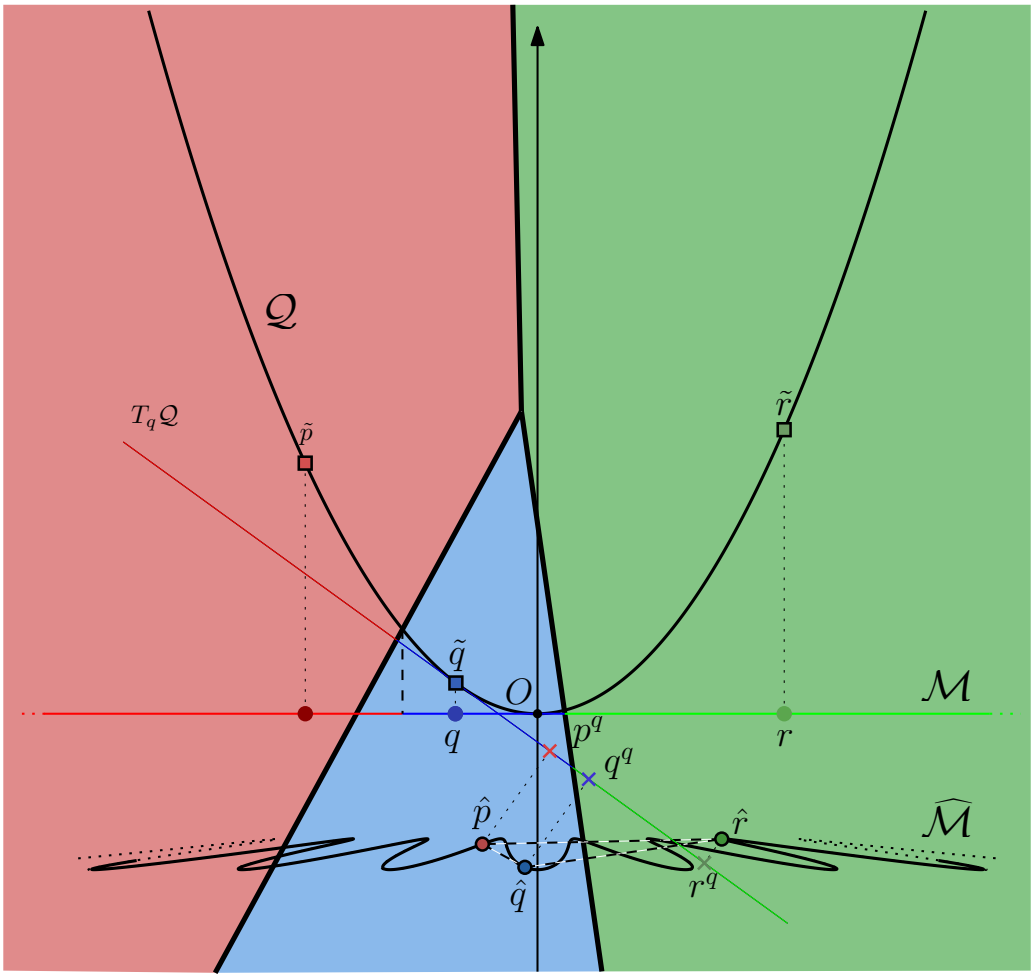

Figure 5.10: Illustration of the notations and concepts ($n = 1$). The colored areas are the power cells of the power diagram. The lines are the bisectors. The metric field is $g(x) = 1 + \cos^2(x)$.

for $p_k \in \mathcal{P}$ with $p_k \neq p_j$. This can be equivalently seen as the definition of the power cell of the seed $\Pi_i(p_h)$ with weight $\omega'_j = \omega_j - \|\widehat{p_j} - \Pi_i(\widehat{p_j})\|^2$, where ω_j is the weight of $\widehat{p_j}$ given by $\omega_j = \|\widehat{p_i}\|^2 - p_i^t M_{p_i} p_i$, in the affine space T_i.

The coordinates of $\Pi_i(\widehat{p_\ell})$ can be obtained by taking the scalar product of $\widehat{p_\ell}$ with a basis $\mathcal{B}_i$ of the tangent space. We compute $\mathcal{B}_i$ through the partial derivatives of the parametrization of the paraboloid (see Section 5.7.1). Figure 5.11 illustrates the anisotropic Voronoi diagram of a set of seeds and the power diagram in two tangent planes.

Projections and transformations

We now prove that the transformations h and t are homeomorphisms, which will be useful in our theoretical analysis of the algorithm.

Lemma 5.7.1 *Both transformation t and h are nice:*

- $h|_{\mathcal{M}}$ *is a homeomorphism from* $\mathcal{M}$ *to* $\widehat{\mathcal{M}}$.
- $t|_{\mathcal{M}}$ *is a homeomorphism from* $\mathcal{M}$ *to* $\widetilde{\mathcal{M}}$.

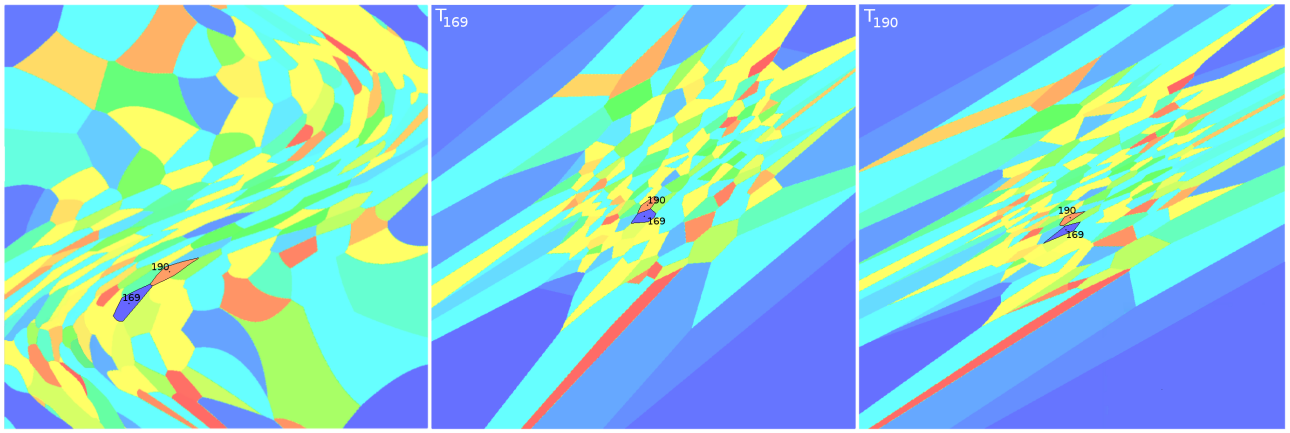

Figure 5.11: Anisotropic Voronoi diagram for the hyperbolic shock metric field (left). Middle and right, the power diagram $\mathrm{PD}_i(\widehat{\mathcal{P}})$ for two adjacent seeds; the nerves are different, creating an inconsistency.

Proof. We prove each property independently.

- The point $h(p)$ is given by $(G_p p, q_p) =: (y_p, y_q)$. The point q_p lives in $\mathbb{R}^{\frac{n(n+1)}{2}}$ and, by construction, its coordinates entirely determine G_p. Once G_p is known, G_p^{-1} can be computed and $p = G_p^{-1} y_p$. On the other hand, it is trivial that if $p_0 \neq p_1$ then $h(p_0) \neq h(p_1)$ because G_p is a positive symmetric definite matrix. Thus h is a bijection between $\mathcal{M}$ and $\widehat{\mathcal{M}}$.

  The metric field g is smooth, therefore all the coordinates of $\hat{x}$ are smooth functions or can be decomposed as sums or products of smooth functions. We need to also prove that the inverse $\Pi_h|_{\widehat{\mathcal{M}}}$ is continuous. Let $\widehat{P} \in \widehat{\mathcal{M}} = (P_1, \ldots, P_D)$. $(P_{n+1}, \ldots, P_D)$ determine fully G_P. Since G_P is a metric (definite positive symmetric matrix), its inverse exists and $\Pi_h(\widehat{P}) = G_P^{-1}.(P_1, \ldots, P_n)$. We can compute G_P^{-1} using the matrix of cofactors. All the coefficients are continuous, and $1/\left|\det(G_P)\right| > 0$ is continuous too. Again, by sums and products of continuous functions, $\Pi_h|_{\widehat{\mathcal{M}}}$ is continuous and $h : \mathcal{M} \to \widehat{\mathcal{M}}$ is a homeomorphism.

- t is a homeomorphism because Π_t is a bi-continuous bijective vertical projection.

□

This proves that $\mathcal{M}$, $\mathcal{Q}$ and $\widehat{\mathcal{M}}$ are homeomorphic.

5.7.2 Refinement algorithms

Along with the alternate construction of the anisotropic Voronoi diagram, Boissonnat et al. [29] proposed a theoretically sound refinement algorithm. This algorithm comes with guarantees in the setting of planar domains, but these guarantees do not extend to higher-dimensional settings.

A refinement algorithm was presented for the tangential Delaunay complex by Boissonnat and Ghosh [24, 17]. This algorithm is relatively similar to the refinement algorithms introduced in Chapter I.4 and in Section 5.4: the refinement point of a simplex σ is chosen as the intersection of the manifold and of the normal space passing through the dual of σ in the tangent space, or through a point close to that dual.

There are two difficulties in the original refinement algorithm. Firstly, the dual of a simplex might intersect the tangent plane, but not the domain itself. Secondly, the Voronoi cell of the refinement

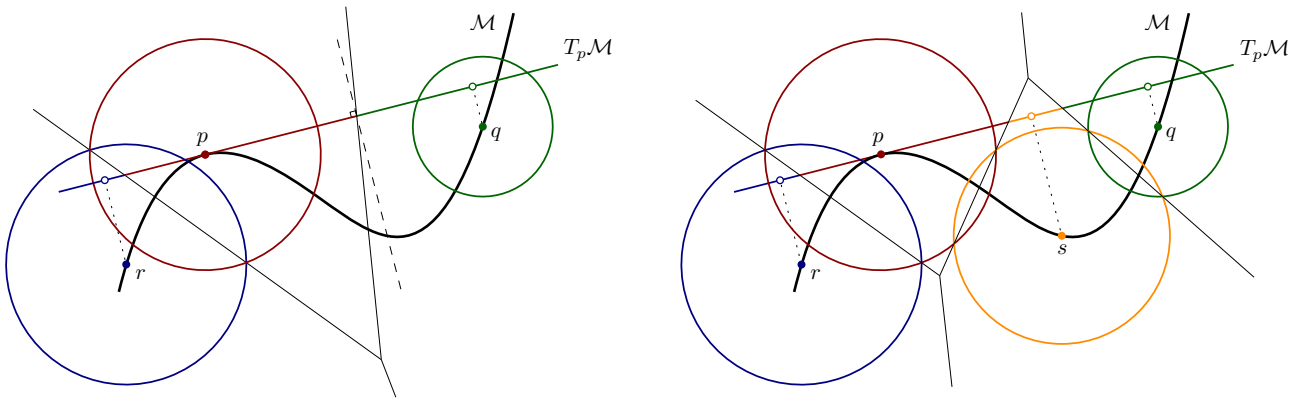

Figure 5.12: The simplex pq in the star S_p is refined by the insertion of its dual and the manifold $\mathcal{M}$. The ambient power diagram is drawn with thin black lines. The circles represent the weights of the points (the radius of the circle is the squared weight).

point might not intersect the tangent plane. Boissonnat and Ghosh show that if the sampling of the point set is sufficiently dense, then those two issues do not appear and the refinement process terminates and produces a good triangulation.

In our context, the normal space does not necessarily approaches the dual of the simplices correctly and the refinement points computed by the original refinement algorithms would not be useful. Instead, we refine bad simplices in the stars by computing the intersection of the dual of the simplex (in the ambient space) with the domain. These computations will be detailed in Section 5.7.5. A theoretical analysis of the refinement algorithm in our context is carried out in Section 5.7.3.

Domains and boundary

We have so far considered domains without boundaries. If we wish to compute a mesh for a bounded domain $\Omega \in \mathbb{R}^n$ (for example a $3D$ surface), it is necessary to know how to compute a refinement point on the domain – and on the boundary. The notion of restricted Delaunay triangulation will once again be applied: the domain and its boundary are lifted to the paraboloid with the map t (see Section 5.5.2) and a simplex is restricted to the domain if the intersection of its dual with the paraboloid is within $t(\Omega)$. If we wish to compute a refinement point on the boundary (for example, to refine only the boundary), the point will be chosen as the intersection of the dual with the boundary lifted onto the paraboloid $\mathcal{Q}$.

5.7.3 Theoretical aspect

We now look towards the theoretical side of our algorithm and would like to prove that the refinement algorithm terminates, that under sufficient conditions the star set is consistent, and that the resulting triangulation is homeomorphic to $\mathcal{Q}$.

We present a detailed road map that could be developed into a complete proof with a fair bit of technical work. We first give an overview of the difficulties that appear due to our modifications of the tangential Delaunay complex.

The difficulties

Two modifications have been made to the original tangential complex algorithm, each complexifying the theory.

Firstly, the ambient diagram is a weighted Voronoi diagram and not a Voronoi diagram. Since the restriction of a power diagram to an affine (tangent) space is a power diagram, this change has no consequence from a construction point of view. However, it is much harder to prove that a refinement point kills the simplex that it is supposed to. Even in the simpler setting where seeds of the ambient power diagram are on the manifold that we are triangulating, it is not clear that a new point should have a weight that is large enough for the new power cell to intersect the tangent space (and thus the simplex is not refined on the tangent plane). This situation is illustrated in Figure 5.13, where the new point (in orange) has a weight so small that its power cell does not intersect the tangent plane. This is however a relatively minor concern: assuming continuity of the weights in the ambient space – which is the case since our weights derive from the metric field, which is continuous – we can show that there exists a lower bound on the sampling such that a refinement point will insert a point that refines the simplex, as done by Boissonnat et al. [18].

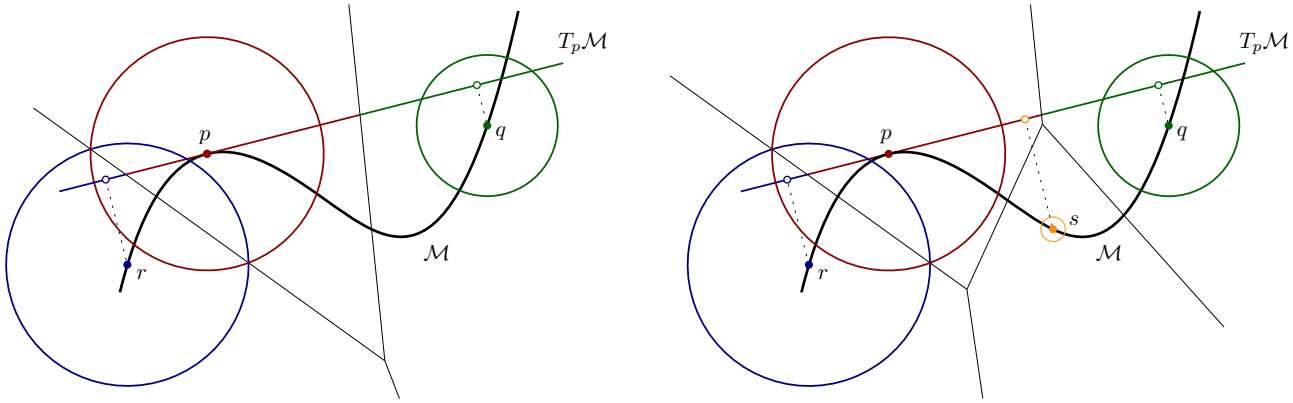

Figure 5.13: The insertion of s (the intersection of the dual of the simplex pq with $\mathcal{M}$) does not refine the simplex pq in the star $T_p\mathcal{M}$. The circles represent the weights of the points (the radius of the circle is the squared weight).

The second modification creates more serious issues. Indeed, we are ultimately interested in a triangulation of the point set $\mathcal{P}$, but our construction builds a star set with vertices $\widehat{\mathcal{P}}$. Admitting that we have obtained a consistent star set, which was merged in a triangulation $\widehat{\mathcal{D}}$, we would like to carry the connectivity of $\widehat{\mathcal{D}}$ onto $\mathcal{P}$. However, while we know that the map Π_h is a homeomorphism between $\widehat{\mathcal{M}}$ and $\mathcal{M}$, the simplices of $\widehat{\mathcal{D}}$ are not necessarily sent nicely onto the simplices obtained by applying the connectivity of $\widehat{\mathcal{D}}$ to the point set $\mathcal{P}$: inversions could be created and the embedding of the dual would subsequently be lost.

Remark 5.7.2 *If we had a triangulation with vertices in $\widetilde{\mathcal{Q}}$ that was homeomorphic to $\mathcal{Q}$ since Π_t is a real projection in the sense that an n-simplex with seeds in $\{\tilde{p}_i\}$ would project onto the n-simplex defined as the convex hull of the $\{p_i\} := \{\Pi_t(\tilde{p}_i)\}$.*

We now present the detailed layout of our proof.

5.7.4 A star-based proof

To prove that the connectivity of the star set brought back to $\mathcal{P}$ is a good triangulation of the input domain, we shall prove that at each point $p \in \mathcal{P}$, the star of p in the star set $\mathcal{S}(\mathcal{P})$ has the same connectivity as the star of p in the dual of the anisotropic Voronoi diagram $\mathrm{Vor}(\mathcal{P})$. We are thus going to almost completely ignore the points $\hat{p}_i$ and $\tilde{p}_i$ and consider the ambient power diagram $\mathrm{PD}(\widehat{\mathcal{P}}_\omega)$ and its restriction $\mathrm{PD}_p(\widehat{\mathcal{P}})$ to the tangent space $T_{\tilde{p}}\mathcal{Q}$.

Consider a point p in $\mathcal{P}$. Let $\{q_i\}$ be the seeds of the Voronoi cells adjacent to $\mathrm{V}_g(p)$ in $\mathrm{Vor}_g(\mathcal{P})$ and let $\mathrm{S}_g^{LS}(p)$ be the star of p in the dual of $\mathrm{Vor}_g(\mathcal{P})$, that is the set of simplices incident to p in the dual of $\mathrm{Vor}(\mathcal{P})$. Denote by $\mathrm{S}_p(p)$ the star of p in the star set $\mathcal{S}(\mathcal{P})$ (that is the star of $\hat{p}$ with connectivity given by $\mathrm{PD}_p(\widehat{\mathcal{P}})$) We thus want to show that $\mathrm{S}_g^{LS}(p)$ and $\mathrm{S}_p(p)$ have the same connectivity.

We shall again make use of power protected point sets. Recall that the support of the dual of a simplex σ is $\mathrm{Aff}(\sigma^\star)$, with $\sigma^\star$ the dual of σ. Our result will stem from the combination of a few observations. Assuming that the sampling is dense, we have that:

- The map h (see Section 5.5.2) is smooth and the image of $\mathcal{M}$ by h (denoted by $\widehat{\mathcal{M}}$ earlier) is a smooth manifold (see Lemma 5.7.1 below). Thus if two points in $\mathcal{P}$ are close, then their lifted equivalents are close in the ambient space.

- The anisotropic Voronoi cells in $\mathrm{Vor}_g(\mathcal{P})$ are not too large. This is a direct consequence of Equation 2.1 as the distortion bounds the difference between the anisotropic distance of Labelle and Shewchuk and the uniform distance using the metric of p. Simplices of the stars $\mathrm{S}_g^{LS}(p)$ and $\mathrm{S}_p(p)$ are thus formed with vertices in $\mathcal{P}$ that are close to p.

- The duals of the maximal simplices of the star $\mathrm{S}_g^{LS}(p)$ (which are $(N-n)$-dimensional power cells in the ambient Voronoi diagram) are roughly parallel to one another because the vertices of the stars are close (and thus are close on $\widehat{\mathcal{M}}$. Note that the support of those duals intersect $\mathcal{Q}$ at the Voronoi vertices of $\mathrm{V}_g(p)$ and, if the dual (and not simply its support) also intersects the tangent plane, the intersections are the Voronoi vertices of the power cell of $\tilde{p}$ in $\mathrm{PD}_p(\mathcal{P})$. Additionally, note that the duals do not become orthogonal to the tangent plane as the sampling increases, but we only need to know that they are more or less parallel from one another.

- The region of interest in the tangent plane (the neighborhood that includes the power cell of $\tilde{p}$ in $\mathrm{PD}_p(\mathcal{P})$) is small and is thus close to the paraboloid.

- The Voronoi cell $\mathrm{V}_g(p)$ of p in $\mathrm{Vor}_g(\mathcal{P})$ is close to the Voronoi cell $\mathrm{V}_{G_p}(p)$. If $\mathcal{P}$ is a power protected net, the Voronoi vertices of $\mathrm{V}_{G_p}(p)$ are separated (see Lemma 3.2.4 in Chapter 3) and thus the Voronoi vertices of $\mathrm{V}_g(p)$ are separated.

The combination of these observations result in the following statement: if $\mathcal{P}$ is a dense power protected net, then the intersections of the supports of the duals of the simplices in $\mathrm{S}_g^{LS}(p)$ with $T_{\tilde{p}}\mathcal{Q}$ are far from one another. Moreover, the orientation is preserved. This is illustrated in Figure 5.14: the supports of the duals of the simplices of the star $\mathrm{S}_g^{LS}(p)$ ("vertical" black lines) are roughly parallel. Since the tangent plane is close to $\mathcal{Q}$ and the Voronoi vertices of $\mathrm{V}_g(p)$ (red circles) are separated, then the red squares (the intersection of the supports of the duals with $T_{\tilde{p}}\mathcal{Q}$) are separated.

Remark 5.7.3 *We have drawn in Figure 5.14 the setting of a two-dimensional paraboloid. By definition, when the intrinsic dimension is $n = 2$, we have an ambient dimension $N = 5$. As we cannot draw five-dimensional objects, we have drawn duals as one-dimensional objects instead of three-dimensional objects. This is a harmless change as the objects of interest are the intersections of the duals with $\mathcal{Q}$ and $T_{\tilde{p}}\mathcal{Q}$ and both are always points (we ignore degenerate cases).*

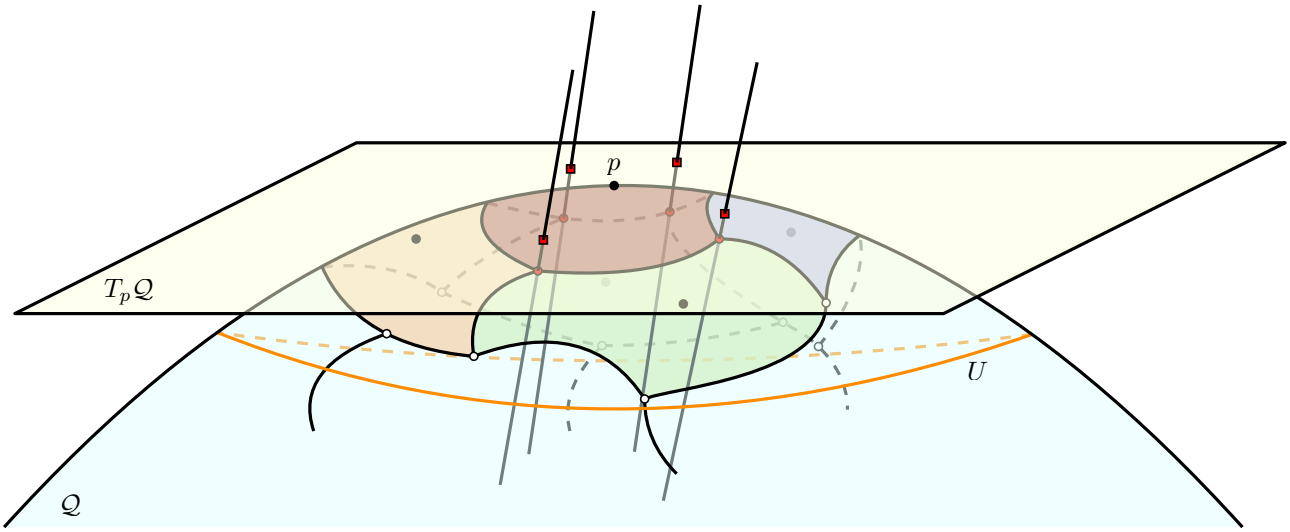

Figure 5.14: The paraboloid $\mathcal{Q}$ and a tangent plane $T_{\tilde{p}}\mathcal{Q}$. The Voronoi vertices of $\mathrm{V}_g(p)$ are marked with red circles and the intersections of the support of the duals (black lines) of the maximal simplices of $\mathrm{S}_g^{LS}(p)$ with $T_{\tilde{p}}\mathcal{Q}$ are marked with red squares. Since the supports of the duals are roughly parallel and the Voronoi vertices are far from one another, the red squares are also far from one another.

It remains to ensure that the duals actually intersect the tangent plane to have the equivalence between the stars. This is not trivial and points arbitrarily far can have power cells that are very close to the paraboloid in ambient space, despite a good sampling of the domain. As we do not have a solution to this issue, we modify slightly our definition of star.

Local stars

We wish to ensure that the dual of a maximal simplex in $\mathrm{S}_g^{LS}(p)$ (the star of p in the dual of the anisotropic Voronoi diagram of Labelle and Shewchuk) also intersects the tangent plane $T_{\tilde{p}}\mathcal{Q}$.

Consider a bisector $\mathcal{BS}$, dual of a maximal simplex of $\mathrm{S}_g^{LS}(p)$. We know that when points are close, they form simplices whose duals are roughly parallel from one another, that the tangent space and the paraboloid are close and that the Voronoi vertices on $\mathcal{Q}$ are separated. For the bisector $\mathcal{BS}$ not to intersect $T_{\tilde{p}}\mathcal{Q}$, it means that it is intersected by another bisector somewhere "between" $\mathcal{Q}$ and $T_{\tilde{p}}\mathcal{Q}$. The simplex corresponding to such dual must involve points that are far from p, otherwise its bisector would be relatively parallel to $\mathcal{BS}$ and since the manifold and the tangent plane are close and the Voronoi vertices are separated, they cannot intersect "between" $\mathcal{Q}$ and $T_{\tilde{p}}\mathcal{Q}$. Thus, if we only build the star $\mathrm{S}_p(p)$ with points that are close to p in $\mathcal{P}$, we ensure that the dual $\mathcal{BS}$ will also intersect $T_{\tilde{p}}\mathcal{Q}$.

Gathering the points close to p can be done through nearest neighbor techniques with respect to the uniform metric field $g_0 = g(p)$, for example.

Conclusion

Using these modified stars, we have that the stars $\mathrm{S}_g^{LS}(p)$ and $\mathrm{S}_p(p)$ have equal combinatorics when the point set is a dense power protected net. Thus if the dual of the anisotropic Voronoi diagram is a triangulation (as done in Section 5.2.1), our star set is also a triangulation.

5.7.5 Computing refinement points

We now investigate the practical side of the algorithm. In their two-dimensional refinement algorithm, Boissonnat et al. [17] compute Voronoi vertices directly in the planar domain as the intersection of the bisectors (conics). This does not extend easily to higher dimensions and we propose alternative methods, based on the alternative construction of the anisotropic Voronoi diagram and our new modified tangential Delaunay complex. In this section, we explain how to compute refinement points of a simplex in any dimension.

Problem statement

Consider a maximal simplex σ, living in a star of $\mathcal{S}(\mathcal{P})$, with vertices $\mathrm{Vert}(\sigma) = \{\widehat{p_i}\}$. Its dual σ^* is the intersection of the power face obtained by intersecting the power cells of the vertices of σ. In our refinement algorithm, the refinement point of σ is, equivalently to a Voronoi vertex in $\mathrm{Vor}_g(\mathcal{P})$, the intersection of a $(N-n)$ power face (σ^*) and the manifold $\mathcal{Q}$. To simplify computations, we firstly consider the intersection of $\mathcal{Q}$ with the support of σ^*, $\mathrm{Aff}(\sigma^*)$. The solutions are then filtered to keep the one(s) that belong to σ^*. The vectors $\overrightarrow{\widehat{p_0}\widehat{p_1}}, \ldots, \overrightarrow{\widehat{p_0}\widehat{p_n}}$ form a basis of the supporting plane of σ, $\mathrm{Aff}(\sigma)$. Denote by c the power center of the simplex σ that lives in $\mathrm{Aff}(\sigma)$. Writing X_i for the i-th coordinate of the point or vector X, the points x in the support of the dual must satisfy :

$$\sum_{i=1}^{N} \left(\overrightarrow{\widehat{p_0}\widehat{p_j}}\right)_i x_i = \sum_{i=1}^{N} \left(\overrightarrow{\widehat{p_0}\widehat{p_j}}\right)_i c_i, \quad \forall j = 1 \ldots n.$$

Each equation simply describes the dual of the edge $[p_0 p_j]$: an hyperplane orthogonal to $[p_0 p_j]$ that passes through c. If we additionally impose that $x \in \mathcal{Q}$, x must satisfy :

$$\sum_{i=1}^{N} \left(\overrightarrow{\widehat{p_0}\widehat{p_j}}\right)_i (t(x))_i = \sum_{i=1}^{N} \left(\overrightarrow{\widehat{p_0}\widehat{p_j}}\right)_i c_i, \quad \forall j = 1 \ldots n. \tag{5.5}$$

This is a non-linear system of n equations and n unknowns. We explain how to solve it in the case of $n = 2$, but this method can be extended to higher dimensions.

Bivariate polynomials Let us rename, for visibility, $\left(\overrightarrow{\widehat{p_0}\widehat{p_1}}\right)_i$ and $\left(\overrightarrow{\widehat{p_0}\widehat{p_2}}\right)_i$ respectively to k_i and ℓ_i, and the right hand side to respectively f_1 and f_2. Proceeding by substitution, the system is given by two equations:

$$\begin{cases} k_1 x_1 + k_2 x_2 + k_3 x_1^2 + k_4 x_1 x_2 + k_5 x_2^2 - f_1 = 0 \\ \ell_1 x_1 + \ell_2 x_2 + \ell_3 x_1^2 + \ell_4 x_1 x_2 + \ell_5 x_2^2 - f_2 = 0 \end{cases} \tag{5.6}$$

Note that we could potentially have $\ell_3 = 0$, and more generally there is no reason for a coefficient – be it a simple k_i, l_i, or a sum and product of k_i and l_i – not to be null.

We solve this system using a resultant-based approach, which is based on Bézout's theorem. The approach of Busé [35] is employed almost entirely as a black box, and we thus do not go in detail. Nevertheless, we specify the necessary matrices below.

The resolution is based on the computation of the generalized eigenvalues and eigenvectors of the companions matrices of the Bézout matrix of the system 5.6. In our case, the matrix can be computed

analytically: the Bézout matrix $BM = (BM)_{i,j}$ is here a 2×2 matrix given by

$$\begin{aligned}
BM_{0,0} &= -k_3\ell_5 x + k_5\ell_3 x - k_2\ell_5 + k_5\ell_2 \\
BM_{0,1} &= -k_4\ell_5 x^2 + k_5\ell_4 x^2 - k_1\ell_5 x + k_5\ell_1 x - k_0\ell_5 + k_5\ell_0 \\
BM_{1,0} &= -k_4\ell_5 x^2 + k_5\ell_4 x^2 - k_1\ell_5 x + k_5\ell_1 x - k_0\ell_5 + k_5\ell_0 \\
BM_{1,1} &= k_3\ell_4 x^3 - k_4\ell_3 x^3 - k_1\ell_3 x^2 + k_2\ell_4 x^2 + k_3\ell_1 x^2 \\
&\quad - k_4\ell_2 x^2 - k_0\ell_3 x - k_1\ell_2 x + k_2\ell_1 x + k_3\ell_0 x - k_0\ell_2 + k_2\ell_0.
\end{aligned}$$

The two companions matrices $A = (A)_{i,j}$ and $B = (B)_{i,j}$ are computed from the Bézout matrix and are 6×6 matrices:

$$A = \begin{pmatrix}
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
a_{4,0} & a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} & a_{4,5} \\
a_{5,0} & a_{5,1} & a_{5,2} & a_{5,3} & a_{5,4} & a_{5,5}
\end{pmatrix},$$

where

$$\begin{aligned}
&a_{4,0} = -k_2\ell_5 + k_5\ell_2, && a_{4,1} = -k_0\ell_5 + k_5\ell_0, && a_{4,2} = -k_3\ell_5 + k_5\ell_3, \\
&a_{4,3} = -k_1\ell_5 + k_5\ell_1, && a_{4,4} = 0, && a_{4,5} = -k_4\ell_5 + k_4\ell_4, \\
&a_{5,0} = -k_0\ell_5 + k_5\ell_0, && a_{5,1} = -k_0\ell_2 + k_2\ell_0, && a_{5,2} = -k_1\ell_5 + k_5\ell_1, \\
&a_{5,3} = -k_0\ell_3 - k_1\ell_2, +k_2\ell_1 + k_3\ell_0 && a_{5,4} = -k_4\ell_5 + k_5\ell_4, && a_{5,5} = -k_1\ell_3 + k_3\ell_1 - k_4\ell_2 + k_2\ell_4,
\end{aligned}$$

and

$$B = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -k_3\ell_4 + k_4\ell_3
\end{pmatrix}$$

Computing the generalized eigenvalues of A and B, that is finding the eigenvalues λ_i and corresponding eigenvectors v_i such that $Av_i = \lambda_i Bv_i$, is the last computational step. Each eigenvalue is then the x-coordinate of a solution of the system. The associated y-coordinate is given – with the matrix expressions above – by the quotient of the last two coordinates of the eigenvector associated to the eigenvalue that we have picked as x-coordinate.

Finally, we test whether the solution is within σ^*.

A simpler approach The approach explained above is complicated and can be prone to numerical errors, which pushed us to seek a simpler method to compute the intersection of the dual with the paraboloid. A first observation is that the intersection of the support of the dual $\mathrm{Aff}(\sigma^*)$ and the paraboloid can be reformulated so that we do not need to know the power center; instead of the system 5.5, we use the system:

$$\|\widetilde{p_0} - t(x)\|^2 - \omega_{\widetilde{p_0}} = \|\widetilde{p_j} - t(x)\|^2 - \omega_{\widetilde{p_j}}, \quad \forall j = 1 \dots n. \tag{5.7}$$

We describe an algorithm based on the system 5.7 to compute an approximation of the intersection. While being simpler, this algorithm also makes use of our tangential complex-based construction.

There are two steps in the algorithm: we first compute the intersection with the dual and the tangent space, and from this first approximation, attempt to find the real one on the manifold $\mathcal{Q}$.

We first consider the intersection of the support of the dual $\mathrm{Aff}(\sigma^*)$ with one of the tangent planes corresponding to a vertex of σ, say $T_{\widetilde{p_0}}\mathcal{Q}$ (noted simply T_0 in the following). We have now a linear system with N equations: the support of the dual is given by the intersection of n hyperplanes and the tangent plane is given by the intersection of $(N-n)$ hyperplanes.

To compute the hyperplanes whose intersection gives T_0, we note that the equation of the tangent plane is given similarly to the first order of a Taylor expansion:

$$f(x) = f(a) + \sum_i \frac{\partial f}{\partial x_i}(a)(x-a)_i.$$

The linearization of the parametrization of $\mathcal{Q}$ gives $(N-n)$ equations and takes two different forms for a given $n \leq k < N$,

- if $x_k = x_i^2$, the linearization is $\widetilde{p_0}_i^2 + 2\widetilde{p_0}_i(x_i - \widetilde{p_0}_i)$;
- if $x_k = x_i x_j$ with $i \neq j$, the linearization is $\widetilde{p_0}_i\widetilde{p_0}_j + \widetilde{p_0}_j(x_i - \widetilde{p_0}_i) + \widetilde{p_0}_i(x_j - \widetilde{p_0}_j)$.

The system possesses thus n equations to express that the intersection belongs to the dual of σ and $(N-n)$ equations to express that the intersection lives on the tangent space. For example, if the intrinsic dimension n is 2, the intersection is given by:

$$\begin{pmatrix} 2(\widehat{p_0} - \widehat{p_1})_1 & \cdots\cdots & 2(\widehat{p_0} - \widehat{p_1})_5 \\ 2(\widehat{p_0} - \widehat{p_2})_1 & \cdots\cdots & 2(\widehat{p_0} - \widehat{p_2})_5 \\ 2\widetilde{p_0}_1 & 0 \quad -1 \quad 0 & 0 \\ \widetilde{p_0}_2 & \widetilde{p_0}_1 \quad 0 \quad -1 & 0 \\ 0 & 2\widetilde{p_0}_2 \quad 0 \quad 0 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} \omega_{\widehat{p_1}} - \omega_{\widehat{p_0}} - \widehat{p_1}^2 + \widehat{p_0}^2 \\ \omega_{\widehat{p_2}} - \omega_{\widehat{p_0}} - \widehat{p_2}^2 + \widehat{p_0}^2 \\ \widetilde{p_0}_1^2 \\ \widetilde{p_0}_1\widetilde{p_0}_2 \\ \widetilde{p_0}_2^2. \end{pmatrix}.$$

This is a linear system, which can be solved easily by conventional means. Note however that it is not guaranteed that the determinant of A should be non-null and it might be necessary to change the tangent plane T_0 to obtain a non-degenerate system. A weighted sums of tangent planes T_i (with $p_i \in \sigma$) can also be used if needed. This is a theoretical concern and this issue never occurs in practice.

Once the system is solved, the solution is a point $s \in \mathbb{R}^N$ and can be moved to $\mathcal{Q}$ by changing its $(N-n)$ last coordinates to match the parametrization. For example, if $n = 2$, we take $s_{\mathcal{Q}} := (x, y, x^2, xy, y^2)$. This approximation can either be kept (but can be really rough), or the real Voronoi vertex can be sought.

Resolution To find the real Voronoi vertex, we solve the system presented in Equation 5.7 with Newton's method, using $s_{\mathcal{Q}}$ as our initial guess. For the system of equations $\mathcal{E}$ of Equation 5.7, the update at each iteration is

$$x \longleftarrow x - J(x)^{-1}H(x),$$

where J is the Jacobian of the system $\mathcal{E}$ (an $n \times n$ matrix).

As for any gradient descent method, one must avoid $\det(J) \approx 0$, which might happen in our case. We monitor the determinant of the Jacobian matrix, and use the classical technique of weighting its diagonal if its determinant becomes too small.

Since the system 5.7 expresses the intersection of $\mathrm{Aff}(\sigma^*)$ – and not σ^* – with $\mathcal{Q}$, we must check that the solution of the system belongs to σ^*, which is always the case in practice.

Inserting a point

We have detailed how to compute efficiently the intersection of the dual of a simplex of the ambient regular Delaunay triangulation with the paraboloid.

The refinement point that we consider in our algorithm is computed from the intersection of the $(N-n)$-flat with $\mathcal{Q}$, or computed similarly to the picking point in the original tangential complex algorithm. This gives a $\tilde{p}$ point on $\mathcal{Q}$, and its corresponding $\widehat{p} = h(\Pi_t(\tilde{p}))$ is added to the seeds $\widehat{\mathcal{P}}$.

5.8 Results

The implementation of our algorithm is similar to our implementation of locally uniform anisotropic meshes (see Section 4.4 in Chapter I.4). A `Star` class represents the local triangulation and a system of refinement queues and mesher levels take care of the refining process.

To compute the star of $\hat{p_i}$ efficiently, we project only a subset of $\mathcal{P}$ of points that are close to $\hat{p_i}$ on the tangent plane T_i and build a local regular triangulation on this plane. This subset is computed through filters to be large enough to ensure that the star of p_i is correct.

As the projected points are still N-dimensional entities, the construction of the n-dimensional regular triangulation on T_i is done by creating local points whose coordinates are the coordinates of $\Pi_i(\widehat{\mathcal{P}})$ in the basis $\mathcal{B}$ chosen for T_i.

5.8.1 Brute force approach

To verify that the computations of duals are correct, we first implement a "brute force" approach that constructs the ambient regular Delaunay triangulation dual of $\mathrm{PD}(\widehat{\mathcal{P}})$ and computes its restriction to the paraboloid $\mathcal{Q}$. The connectivity of the restriction of the $\mathrm{Del}(\widehat{\mathcal{P}})$ applied to $\mathcal{P}$ is expected to be the exact the dual of the anisotropic Voronoi diagram Vor_g. Note however that the restriction to $\mathcal{Q}$ is a triangulation embedded in $\mathbb{R}^N$, and it is the action of transferring its connectivity to $\mathcal{P}$ that creates inversions. By nature, the tangential complex introduces inconsistencies, which are – with inversions – another type of problematic configurations that prevent the . Computing the brute force result is thus also a good way to observe issues for a given point set that are not linked to the use of the tangential complex.

Regular Delaunay triangulations in arbitrary dimension are available in the CGAL [2] library. For planar cases ($n = 2$), the dimension of the ambient space is 5 and thus the computational time is still reasonable (as long as the number of vertices is not too large). Additionally, while $\widehat{\mathcal{P}}$ lives in $\mathbb{R}^N$, the regular triangulation's dimension can have an intrinsic dimension lower than N as $\widehat{\mathcal{P}}$ can live on a k-flat (with $n \leq k < N$); for example, if the metric is uniform, then the $N-n$ last coefficients of all the points in $\widehat{\mathcal{P}}$ are identical.

We firstly compare the anisotropic Voronoi diagram for the distance of Labelle & Shewchuk with the result obtained through the brute force implementation. A random point set is generated over a square endowed with the metric field defined at a point $p(x, y)$ by $M(x, y) = \begin{pmatrix} \exp(x) & 0 \\ 0 & 1 \end{pmatrix}$. This point set is intentionally sparse to create orphans and degenerate configurations. The result is shown in Figure 5.15; as expected, the two structures are dual.

5.8.2 Tangential complex results

We now present some results obtained with the star set based refinement algorithm. Due to the issues previously mentioned (see Section 5.7.3 and Figure 5.13), the refinement algorithm will often

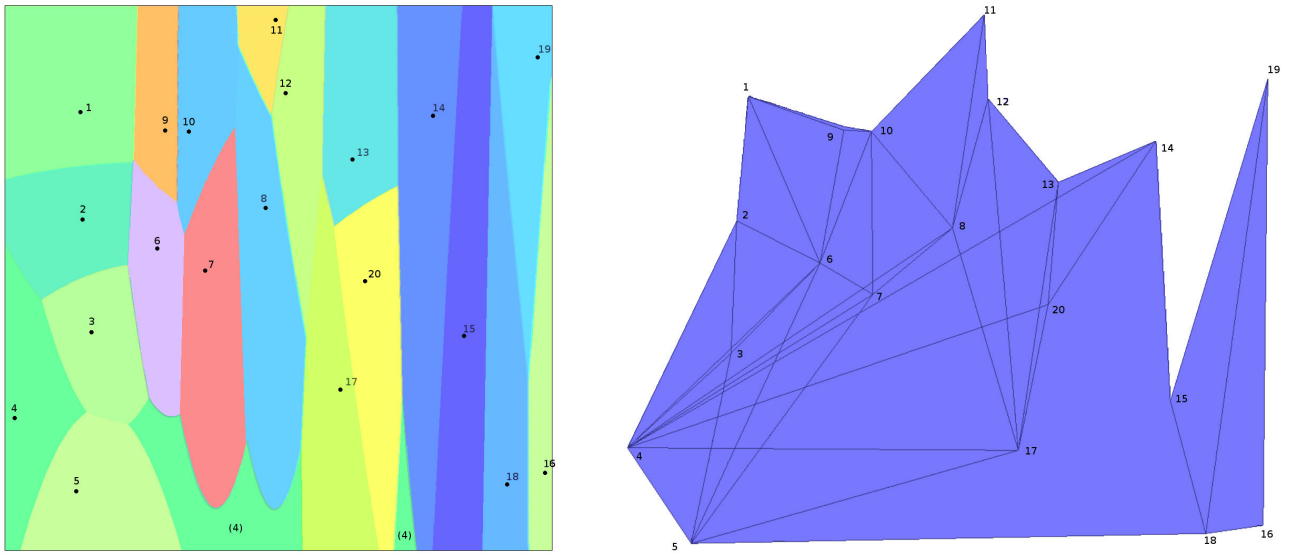

Figure 5.15: Anisotropic Voronoi diagram of a set of seeds (left) and the restriction of the high-dimensional regular Delaunay triangulation to the paraboloid $\mathcal{Q}$ (right).

fail to destroy simplices in the tangent planes if the sampling is not already dense. We thus start with relatively dense point sets, generated by our implementation of the locally uniform anisotropic framework (see Chapter I.4) for the same domain and metric field.

Figure 5.16 shows the star set obtained for a square endowed with two different metric fields, a two-dimensional version of the unidimensional shock and the hyperbolic shock metric fields (see Sections A.7 and A.3 in Appendix A). These two examples are sufficient to obtain a good idea of the potential of the method as they exhibit various anisotropic configurations. In the case of the unidimensional shock metric field (left, in the figure), the star set was initially inconsistent, but was refined and eventually achieved consistency. Similarly to locally uniform anisotropic meshes, many insertions were needed to solve inconsistencies. The result on the right in Figure 5.16 is left unrefined to show the state of the star set after the insertion of the initial vertices. Despite a relatively dense and well positioned point set, the star set contains many inconsistencies in regions where the algorithm described in Chapter I.4 also had issues.

In conclusion, this algorithm suffers unfortunately from the same issues that were encountered in the empirical study of the locally uniform anisotropic mesh framework, in Chapter I.4: consistency is hard to achieve between the stars of the tangential complex and many insertions are required. Similar issues were observed using point sets generated with other anisotropic mesh generators.

As for locally uniform anisotropic meshes, we attempted to improve the resolution of inconsistencies from other means than point insertion (through the perturbation of the weight and position of vertices, for example), but with no success here either.

5.9 Conclusion

Our interest for the anisotropic Voronoi diagram introduced by Labelle and Shewchuk stemmed from the its simple definition and computations.

In this chapter, we have introduced requirements on a set of seeds which guarantee that the dual of the anisotropic Voronoi diagram built from these seeds is a triangulation. We require that the seed

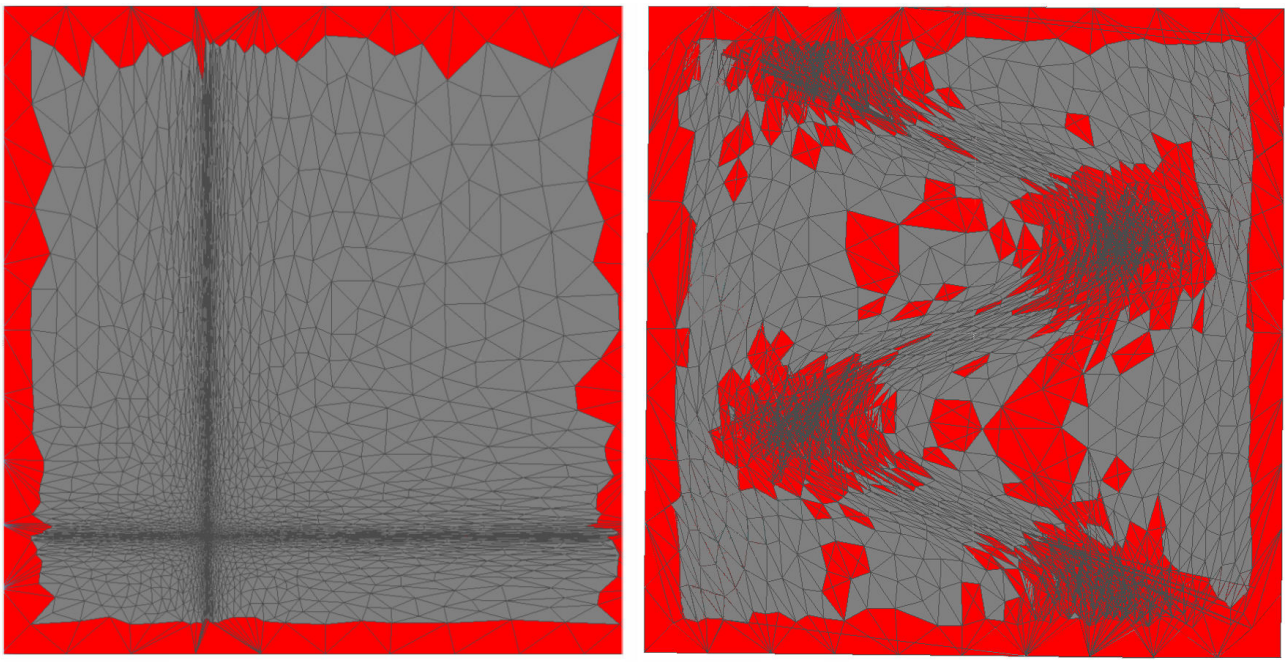

Figure 5.16: Two results obtained with the tangential complex. On the left, a result obtained by refining a point set; a consistent star set is obtained. On the right, the inconsistencies (in red) in the star set before the refinement algorithm finishes.

set forms a net of the domain, meaning that it is a good sampling of the domain and the seeds are not too close from one another. More importantly, we require that the seed set is power protected. Indeed, a consequence of power protection is that the Voronoi vertices are separated, allowing us to locally approximate the anisotropic Voronoi diagram of Labelle and Shewchuk with a Voronoi diagram built using a uniform metric field. We presented alongside a refinement algorithm to construct power protected nets and exposed the similarities between power-protection and quasi-cosphericities. The construction of anisotropic Voronoi diagrams from actual power protected nets was not investigated in practice and this should be the topic of future research. One could hope that the theoretical requirements do not need to be met in practice.

In a second part, we have introduced an alternative way to construct the dual of the Labelle and Shewchuk's Voronoi diagram using the tangential Delaunay complex. The modifications brought to the original Delaunay triangulation, and specifically the fact that the seeds of the ambient diagram do not belong to the manifold, make it hard to extend the original theory. The issues preventing the theory from being extended are well known and their resolution will be the topic of future work. In practice, this approach yields results that are similar to those of Chapter I.4: meshes that honor the metric field, but are over-refined. As for locally uniform anisotropic meshes, this is not satisfying.

In the following chapters, we shall completely abandon star set-based approaches and take interest in the Riemannian Voronoi diagram. Often considered as too expensive to compute, we propose a new discrete structure that captures its nerve. Our algorithm is theoretically sound and some positive results are obtained.

Part III

Riemannian Voronoi diagrams

6. Discrete Riemannian Voronoi diagrams

Several authors have considered Voronoi diagrams based on anisotropic distances to obtain triangulations adapted to an anisotropic metric field. These authors hoped to build upon the well-established concepts of the Euclidean Voronoi diagram and its dual structure, the Delaunay triangulation, for which many theoretical and practical results are known [12].

The Voronoi diagram of a set of points $\mathcal{P} = \{p_i\}$ in a domain Ω with respect to a distance d is a partition of Ω in a set of regions $\{\mathrm{V}_d(p_i)\}$. The region $\mathrm{V}(p_i)$ is called the Voronoi cell of the seed p_i and is defined by

$$\mathrm{V}(p_i) = \{x \in \Omega \mid d(p_i, x) \leq d(p_j, x),\ \forall p_j \in \mathcal{P} \backslash p_i\}.$$

Du and Wang [64] defined the anisotropic Voronoi cell of a point p_i in a domain Ω endowed with a metric field g as

$$\mathrm{V}_g^{DW}(p_i) = \{x \in \Omega \mid d_{g(x)}(p_i, x) \leq d_{g(x)}(p_j, x),\ \forall p_j \in \mathcal{P} \backslash p_i\}.$$

Canas and Gortler [37, 38] and Cheng et al. [50] proved that under sufficient sampling conditions, the dual of this Voronoi diagram is a triangulation for planar and surface domains. With the same idea, Labelle and Shewchuk [89] introduced slightly different Voronoi cells:

$$\mathrm{V}_g^{LS}(p_i) = \{x \in \Omega \mid d_{g(p_i)}(p_i, x) \leq d_{g(p_j)}(p_j, x),\ \forall p_j \in \mathcal{P} \backslash p_i\}.$$

While the diagram obtained is easier to trace since bisectors are quadrics, the associated theory only covers the planar setting and its extension to higher dimensions faces inherent difficulties. In Chapter II.5, we have shown that this theory could be extended to any dimension using a specific type of point sets, power protected nets, and proposed a refinement algorithm to generate such point sets.

Both these diagrams aim to approximate the Riemannian Voronoi diagram (Vor) whose cells are defined by

$$\mathrm{V}_g(p_i) = \{x \in \Omega \mid d_g(p_i, x) \leq d_g(p_j, x), \forall p_j \in \mathcal{P} \backslash p_i\}.$$

where $d_g(p, q)$ is the geodesic distance on the Riemannian manifold. Riemannian Voronoi diagrams have been previously considered when the surface is endowed with the natural Euclidean metric field. Peyré and Cohen [115] use isotropic Riemannian Voronoi diagrams on surfaces and devise a farthest point refinement algorithm to achieve isotropic adaptive remeshing. Similarly, Liu et al. [99] employ the Riemannian Voronoi diagram to obtain the intrinsic Delaunay triangulation of a surface. Riemannian Voronoi diagrams with polyline sources are considered by Xu et a. [141] and methods to draw three-dimensional Riemannian Voronoi diagrams are introduced by Naß [108].

Judged too expensive to consider, the Riemannian Voronoi diagram of a manifold endowed with a metric field has not been the subject of any practical work. However,the Riemannian Voronoi diagram comes with the benefit of providing a more favorable framework to develop a theoretically sound method. For example, it is devoid of orphan cells, which are regions of the domain disconnected from their seed. These orphans, who might prevent the dual from being a triangulation, appear in the other anisotropic Voronoi diagrams previously mentioned. Finally, the Riemannian Voronoi diagram is by definition the diagram that captures best the variations of metrics and we wish to study the quality of the approximation of other anisotropic Voronoi diagrams.

We approach the Riemannian Voronoi diagram and its dual with a focus on both practicality and theoretical robustness and introduce an algorithm to compute a discrete Riemannian Voronoi diagram. The idea of building Voronoi diagrams in a discrete manner has been studied before in the isotropic setting by Hoff [78]. Cao et al. [39] studied the digital Voronoi diagram, a discrete structure diagram obtained by flooding from the seed and proved that they obtain a geometrically and topologically correct dual. Their work is however limited to the 2D isotropic setting.

The theoretical study of the algorithm will be the topic of Chapters III.7 and III.8.

Contributions This chapter presents a first step towards practical Riemannian Voronoi diagrams. We introduce a novel structure to approximate Riemannian Voronoi diagrams, along with a refinement algorithm that is generic with respect to the choice of the method used to compute geodesic distances.

We obtain a significant speed up over standard geodesic distance computations by using locally uniform anisotropic meshes (presented in Chapter I.4) as the underlying structure upon which geodesic distances are computed. Additionally, our method produces meshes that do not suffer from the over-refinement that was observed in Chapters I.4 and II.5. Consequently, we obtain thus an algorithm that produces good results (albeit with some other defaults) and is provable, as shall be seen in Chapters III.7 and III.8. The structure and the implementation are described and the practicality is investigated.

Contents

6.1 The discrete Riemannian Voronoi diagram

The computation of geodesic path lengths in any domain is a difficult task as there is generally no closed form available. The wide range of domain can be shrunk, through mesh generation, to consider only piecewise-linear domains. Even in this simpler setting, the computation of geodesic distances and paths is still a complex problem to which much work has been dedicated in the last decades. Despite many studies, geodesic distances still cannot be obtained exactly for most domains endowed with an arbitrary metric field. Nevertheless, we introduce a discrete structure that is, under some conditions, combinatorially equivalent to the Riemannian Voronoi diagram and whose duals are triangulations.

6.1.1 Computation of geodesic distances

Historically, geodesic distances were first evaluated with respect to the Euclidean metric naturally derived from the embedding of the domain in $\mathbb{R}^n$. In this context, many approaches are available depending on the expectations on the geodesic distance: exact or approximate, heuristical or with theoretical guarantees, ... (see Peyré [116]).

Algorithms have more recently been introduced to compute geodesic distances when the domain is endowed with a prescribed metric field that exhibits anisotropy. We describe some of these methods, which we have empirically evaluated.

Konukoglu et al. [88] proposed a heuristical method to extend the fast marching method to the anisotropic setting by inserting a recursive correction scheme in the standard algorithm. However, this causes the loss of theoretical convergence of the fast marching method and the added recursivity induces a heavy computational cost. Mirebeau [105] introduced a grid-based variation of the fast marching method. The algorithm comes with theoretical guarantees, but the underlying grid structure does not extend easily to manifolds. A very different approach was considered by Crane et al. [55], who solve linear elliptic equations to compute a global geodesic distance map, obtaining an approximate distance between any two points of the domain. The algorithm is much faster than front propagation-based methods, but its parameters require delicate tuning and the approximation is generally much worse, making it unattractive for our work. Campen et al. [36] proposed a heuristical approach based on a modified Dijkstra algorithm, which we shall use, and describe in more detail in the section.

Short-term vector Dijkstra

The algorithm of Campen et al. [36] is inspired by the concept of *vector-Dijkstra* introduced by Schmidt et al. [125], who measure the geodesic distance between the endpoints of a path $\mathcal{L}$ (composed of edges of a triangulation) as the length of the segment e between the endpoints of the path, rather than the sum of the lengths of the edges $\{l_i\}$ that compose the path. This technique is adapted to the anisotropic setting by Campen et al. by subdividing the segment e between the endpoints of the path into smaller segments $\{e_i\}$ whose length is computed in a metric. Specifically, the subsegments $\{e_i\}$ are obtained by projecting the edges l_i of the path $\mathcal{L}$ onto the segment e. The length of e is then defined as the sum of the lengths of the edges e_i computed with the metric l_i. This process is illustrated in Figure 6.1.

This method of estimating lengths is then combined with a traditional Dijkstra approach, with the nuance that the distance at the neighbor n of a fixed point p is computed not by simply adding the

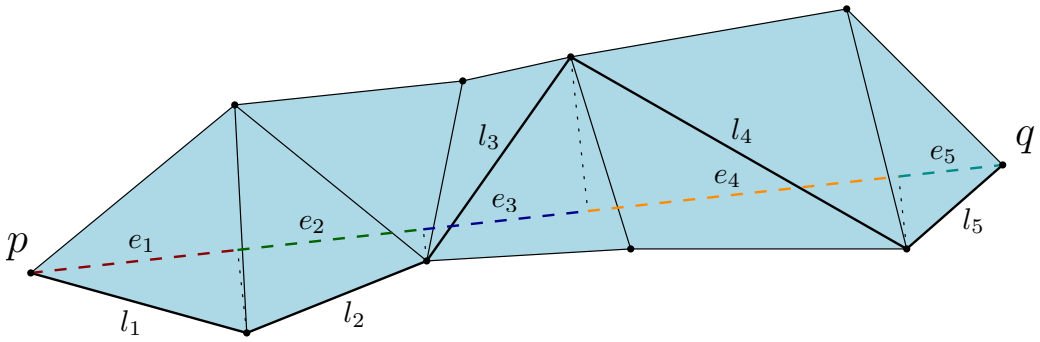

Figure 6.1: Computation of the distance between the vertices p and q. The triangulation is drawn with black edges and the path between p and q with thickened black edges.

distance $d(n, p)$ to the distance between p and the source s, but by finding the point p_k that minimizes the sum $d(n, p_k) + d(p_k, s)$, where $d(n, p_k)$ is computed with the vector approach explained above. Additionally, not every p_k is eligible to be the minimizer: p_k must be at most the k-th ancestor of p, where an *ancestor* of a point p is the neighbor of p from which the distance between p and its closest seed is minimum (and recursively, a k-th ancestor of p is the ancestor of the $(k-1)$-th ancestor of p). The value of k is a parameter of the algorithm and must be appropriately chosen.

Although the short-term vector Dijkstra algorithm does not offer guarantees, it is nevertheless fast and precise, with little requirement on the quality of the discrete domain. This method additionally is well defined for surfaces, and does not require an embedded triangulation to compute geodesic distances, which will be particularly useful in our work. For these reasons, we shall principally use this method to compute geodesics.

6.1.2 Canvas

We refer to the discretization of the domain Ω upon which geodesic distances are computed as the *canvas*, and denote it $\mathcal{C}$. Notions related to the canvas will explicitly carry *canvas* in the name (for example, an edge of $\mathcal{C}$ is a *canvas edge*). By default, the canvas is assumed to be a good isotropic triangulation of the domain, based on the notion of restricted Delaunay triangulations (see Section 2.11 in Chapter 2). From Section 6.2 on, we will also make use of anisotropic canvasses, but we assume that the canvas is an isotropic triangulation until then. The generation of canvasses is detailed in Section 6.1.6.

6.1.3 Generation of the Discrete Riemannian Voronoi Diagram

Extending the metaphor, we now describe the process of "painting" the canvas, that is the attribution of a color – which corresponds to a seed of the diagram – to all of its vertices. We assume in this section that the canvas and the seed set are known and fixed. The generation of the set of seeds will be the topic of Section 6.1.7. The canvas is colored with a process that is similar to the classic Dijkstra algorithm, which we now describe.

Initialization

Three fields are associated to each vertex of the canvas:

- a color: the color of the closest Voronoi seed (represented with an integer and initialized at -1).

- a distance: the current smallest geodesic distance to a seed (initialized at ∞),
- a status: KNOWN, TRIAL or FAR (initialized at FAR),

The initialization of the canvas is described in Algorithm 15. It is assumed that the canvas is sufficiently dense for multiple seeds to not fall within the same canvas simplex.

done by locating each seed on the canvas, that is

Algorithm 15 Initialization of the canvas

```
for each seed s_i ∈ P do
    Find a canvas simplex σ_C which contains the seed
    for each vertex pof σ_C do
        Set the status of p to TRIAL
        Set the color to the color of the seed s_i
        Set the distance to the distance to the seed
```

Remark 6.1.1 *If a seed belongs to the boundary of the domain, it is possible that the seed does not fall within an interior canvas simplex. In this case, it will nevertheless fall within a surface Delaunay ball and the canvas simplex whose surface Delaunay ball contains the seed is chosen as the canvas simplex that "contains" the seed.*

Propagation

Colors are spread from the vertices of the canvas simplices that were initialized to the rest of the canvas through simultaneous front propagations. For this purpose, TRIAL canvas vertices are kept in a priority queue $\mathcal{R}$ that is sorted by increasing distance to the vertex's closest seed. At each iteration, the vertex v with lowest distance to a seed p is removed from the queue and its status is changed to KNOWN. For each neighbor n of v whose status is not KNOWN, the distance d to the seed p is evaluated. If d is smaller than the current distance d_{old} between n and its current closest seed p_{old}, the fields of n are modified accordingly (the status becomes TRIAL, the color that of p, the distance is set to d), and the priority queue $\mathcal{R}$ is updated. The full procedure is detailed in Algorithm 16.

Algorithm 16 Computation of the discrete Riemannian Voronoi diagram

```
Initialize the canvas with default values
for each seed s_i ∈ P do
    Initialize the canvas with the seed s_i (Algorithm 15)
while R is not empty do
    Pick the TRIAL vertex p with smallest distance to a seed s_p
    Remove p from R
    Mark p as KNOWN
    for each vertex n neighbor of p with status not KNOWN do
        Compute the geodesic distance d at n from s_p
        if d is smaller than the current distance at n then
            Color n with the color of p
            Assign d as the distance at n
            Mark n as TRIAL and insert it (or update its position) in R
```

All the geodesic distance computation algorithms described in Section 6.1.1 can be adapted for use in Algorithm 16, with varying degrees of difficulty and practicality to compute the discrete Riemannian Voronoi diagram.

Termination

Since one vertex is switched from TRIAL to KNOWN every iteration, the queue is eventually empty and the vertices of the canvas are all colored. An example of fully colored canvas is exposed in Figure 6.2.

From the completely colored canvas, we build our discrete Riemannian Voronoi diagram, defined below.

Definition 6.1.2 (Discrete Riemannian Voronoi diagram) *Consider a point set $\mathcal{P}$ and a canvas $\mathcal{C}$. To each seed p_i, we associate its* discrete cell $\mathrm{V}_g^d(p_i)$ *defined as the union of all canvas simplices with at least one vertex of the color of p_i. We call the set of these cells the* discrete Riemannian Voronoi diagram *of $\mathcal{P}$, and denote it by* $\mathrm{Vor}^d(\mathcal{P})$.

Note that, by construction, a vertex can only take the color of one of its KNOWN neighbors, and thus a discrete cell is connected. Observe additionally that, contrary to typical Voronoi diagrams, our discrete Riemannian Voronoi diagram is not a partition of the domain. Indeed, there is a 1 canvas simplex-thick overlapping due to each canvas simplex belonging to all the Voronoi cells whose color appears in its vertices. This is intended and allows for a natural definition of the nerve of this diagram (see Section 6.1.4).

6.1.4 Extraction of the nerve of the discrete diagram

Since our main interest resides in anisotropic triangulations, we are interested in the dual of discrete Riemannian Voronoi diagrams. To construct this dual, the combinatorial information of the diagram must first be extracted from the canvas.

The *nerve* of the discrete Riemannian Voronoi diagram is given by

$$\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P})) = \left\{ \{\mathrm{V}_g^d(p_i)\} \mid \cap_i \mathrm{V}_g^d(p_i) \neq \emptyset, p_i \in \mathcal{P} \right\}.$$

Using a triangulation as canvas offers a very natural way to construct the nerve $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P}))$ since each canvas k-simplex σ of $\mathcal{C}$ has $k+1$ vertices $\{v_0, \ldots, v_k\}$ with respective colors $\{c_0, \ldots, c_k\}$, which correspond to seeds $\{p_{c_0}, \ldots, p_{c_k}\} \in \mathcal{P}$. Indeed, a canvas σ simplex belongs to each Voronoi cell whose color appears in the vertices of σ. The intersection of the Voronoi cells whose color appears in the vertices of σ is thus non-empty and the set D of these Voronoi cells belongs to the nerve. In that case, we say that the simplex σ *witnesses* (or is a witness of) D. By extension, if $(\mathcal{K}, \varphi)$ is a realization of $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P}))$, we also say that σ witnesses the simplex $\varphi(D)$. For example, if the vertices of a 3-simplex τ have colors red–blue–blue–red, then the intersection of the discrete Voronoi cells of the seeds p_{red} and p_{blue} is non-empty and $D := \{\mathrm{V}_g^d(blue), \mathrm{V}_g^d(red)\}$ belongs to $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P}))$. The simplex τ thus witnesses the (abstract, for now) edge between p_{red} and p_{blue}.

For all $D \in \mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P}))$, if there exists $\sigma \in \mathcal{C}$ such that σ witnesses $\varphi(D)$, and $\forall \sigma \in \mathcal{C}$ the simplex witnessed by σ belongs to $\varphi(\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P})))$ then we say that $\mathcal{C}$ *captures* $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P}))$. A Voronoi cell is said to be *captured* if all its Voronoi vertices are witnessed by the canvas. Figure 6.2 illustrates a canvas, the discrete Voronoi cells, and the notable witnesses of the nerve.

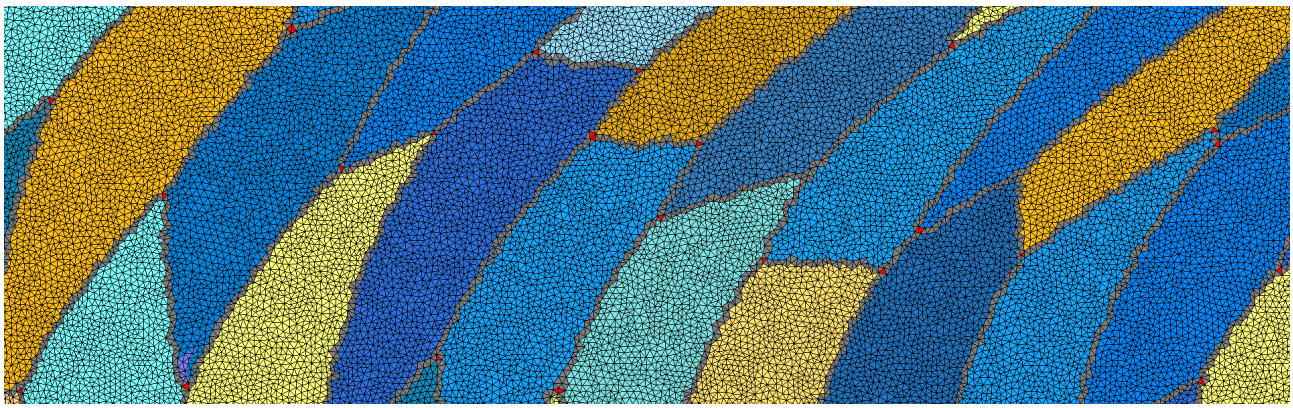

Figure 6.2: The canvas simplices colored in red are witnesses of Voronoi vertices. The canvas simplices colored in grey are witnesses of Voronoi edges. Canvas simplices whose vertices all have the same color are colored with that color.

Remark 6.1.3 *If* $\bigcap_{i=0\dots k} \mathrm{V}_g^d(p_{c_i})$ *is non-empty, then the intersection of any subset of* $\{\mathrm{V}_g^d(p_{c_i})\}_{i=0\dots k}$ *is non-empty. In other words, if a canvas simplex* σ *witnesses a simplex* $\varphi(D)$*, then for each face* τ *of* $\varphi(D)$*, there exists a face of* σ *that witnesses* τ*. This implies that, assuming that the dual of the Riemannian Voronoi diagram is pure, it is sufficient to only consider maximal canvas simplices whose vertices have all different colors to build* $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P}))$*. As we shall use this remark for an unbounded domain endowed with an Euclidean metric field, this assumption will be satisfied.*

6.1.5 Geometric realizations of the nerve

Once the nerve of the diagram is known, a mesh can be obtained by tracing classical simplices (defined as the convex hull of the vertices) between the seeds, accordingly to the combinatorial information of the nerve. This yields the *straight Riemannian Delaunay triangulation*, which we denote by $\overline{\mathrm{Del}}_g(\mathcal{P})$.

Although we are primarily focused on the generation of "straight" anisotropic triangulations, it should be noted that we have constructed an approximation of the Riemannian Voronoi diagram, and the curved realization of the dual is in a way more natural since the diagram is after all computed using geodesic distances.

Curved Riemannian Delaunay triangulations (see Section 2.9 in Chapter 2) have received some theoretical attention [91, 66, 65], but there is no work on the practical side of the subject. To investigate this practical side, we are required to know how to trace curved Riemannian simplices. In the case of planar and surface domains, simply computing the geodesic paths between the vertices of a simplex provides the complete curved Riemannian simplex. However, drawing curved Riemannian simplices in higher dimensions is far more complicated since curved Riemannian simplices are defined through a minimization process at each point.

We will consider both realizations of the nerve in our theoretical and practical studies.

6.1.6 Generation of the canvas

We now detail the process of generating the canvas, the underlying structure upon which geodesics are computed.

Observe that if the canvas is too sparse or badly approximates the domain, the discrete Riemannian Voronoi diagram of a set of seeds might be combinatorially notably different from the Riemannian

Voronoi diagram, as illustrated for example in Figure 6.3. Our algorithm aims to be a discrete yet accurate version of the Riemannian Voronoi diagram. The canvas must therefore be adapted to our input domain, metric field and set of seeds.

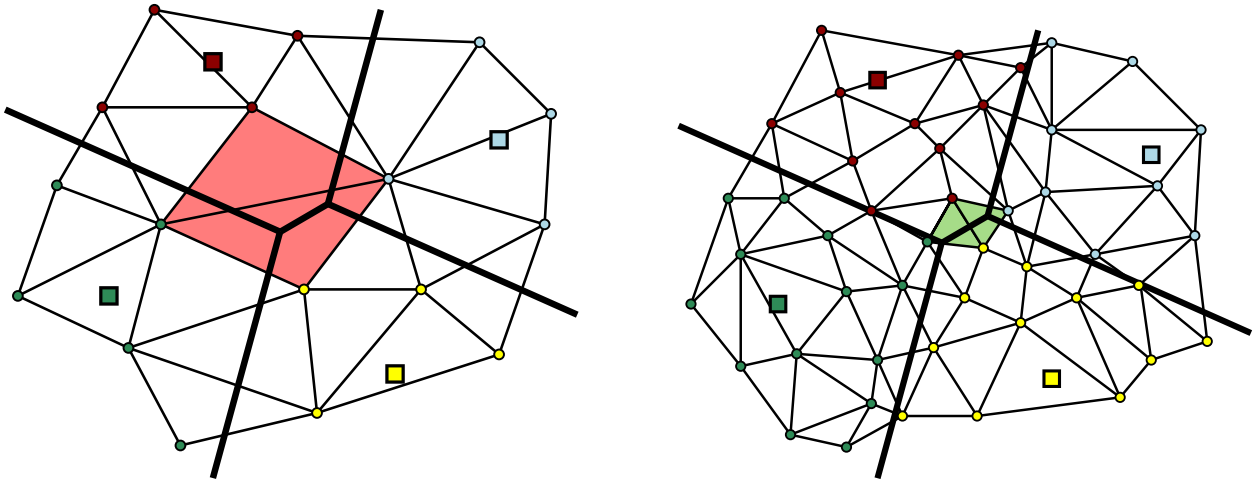

Figure 6.3: Two discrete Riemannian Voronoi diagram in the simple case of a plane endowed with the Euclidean metric field. The seeds are drawn with squares, the canvas with thin black segments and the canvas vertices with circles. The Riemannian Voronoi diagram is drawn with thick black segments. On the left, the canvas is not dense enough to capture the combinatorial information of the Riemannian Voronoi diagram.

For now, we limit the nature of the canvas to an isotropic triangulation of the domain, generated by a classical isotropic refinement Delaunay algorithm that is driven by a sizing constraint. The main unknown is the necessary value of sizing field for the canvas to capture the nerve of the Riemannian Voronoi diagram. To simplify matters, we assume that the sizing field is constant.

Two possibilities arise, depending the knowledge of the set of seeds.

Known seeds

Firstly, consider the case where the seeds are known before the generation of the canvas. The sizing field of the canvas must then be a factor smaller than the minimum distance between two Voronoi vertices of the Riemannian Voronoi diagram. Although this value could be computed, seeds are almost never known beforehand: using seeds obtained through simpler (to compute) diagrams, such as Labelle and Shewchuk's, or from other anisotropic methods (for example using a locally uniform anisotropic mesh) do not in practice yield "good" Riemannian Voronoi diagrams in term of embeddability of the dual nor in term of quality of the simplices. Therefore, we skip the computation of the required sizing field of the canvas in this (pointless) case.

Unknown seeds

The more complicated case, but also the realistic one, is that we do not know the seeds in advance and use the canvas to generate well-positioned seeds. The canvas must thus be generated preemptively and be adapted to the set of seeds that is going to be generated upon this very same canvas.

In theory In Chapters III.7 and III.8, we shall prove that if the set of seeds is a δ-power protected (ε, μ)-net (see Sections 2.12.1 and 2.12.2 in Chapter 2 for the respective definitions), then we can compute an upper bound on the sizing field of the canvas such that the canvas captures the combinatorial information of the Riemannian Voronoi diagram if the sizing field is honored. A rough understanding of these requirements is that the net assumption gives us a dense and separated point set, and the power protection ensures that the Voronoi vertices are far from one another, making it possible to compute a bound on the sizing field of the canvas such that issues like the one illustrated in Figure 6.3 do not happen.

In practice The theoretical upper bound on the sizing field of the canvas is extremely small, but thankfully does not need to be honored in practice. In practice, we estimate the required sizing field by computing the smallest potential edge length, as detailed below.

Similarly to the generation of the canvas, the generation of our anisotropic triangulations is driven by a sizing constraint, with the purpose of creating meshes whose edge lengths are close to unit (see Section 2.13 in Chapter 2). The seed set thus honors a sizing field with a constant parameter r_0, meaning that the shortest path between a seed and a Voronoi vertex is at most r_0. In the discrete context, this means that the distance between any vertex of a canvas simplex witnessing a Voronoi vertex of the Riemannian Voronoi diagram and the corresponding closest seed it smaller than r_0. As explained in Section 2.13 in Chapter 2, it is more convenient to encode the sizing field directly into the metric field and we thus always use $r_0 = 1$.

If the seeds are well spread, there exists a lower bound ℓ on the distance between two seeds. By definition, a unit ball in the metric G_p at a point p is an ellipsoid with semi-axis lengths $1/\sqrt{\lambda_p^i}$, thus the smallest distance (in Euclidean space) between a seed and a point on the unit ball is given by $\min_i 1/\sqrt{\lambda_p^i} = 1/\sqrt{\max_i \lambda_p^i}$. The sizing field of the canvas is then taken as:

$$h_{\mathcal{C}} = \min_{p \in \mathcal{P}} \frac{\ell}{10\sqrt{\max_i \lambda_p^i}}.$$

This ensures that we have at least 10 canvas simplices between two Voronoi seeds, which is generally enough to guarantee that the Riemannian Voronoi diagram of the set of seeds is captured. The generation of the seeds is the topic of the next section.

Remark 6.1.4 *As we need to compute accurate geodesic distances, the sizing field should also be sufficiently low with respect to metric distortion. If the canvas is extremely sparse, the sizing field will be thus determined by this criterion. However, in the common case, this constraint is automatically satisfied because the set of seeds is sufficiently dense with respect to metric distortion.*

6.1.7 Generation of the seeds

For the reasons explained in the previous section, we generally do not know the seeds beforehand.

In theory Our theoretical analysis of our discrete structure (in Chapters III.7 and III.8) shows that if the set of seeds is a δ-power protected (ε, μ)-net (with known bounds on δ and ε), then the discrete diagram has exactly the same combinatorial structure as the (exact) Riemannian Voronoi diagram, as long as the canvas is dense enough. We have presented in Section 5.4 in Chapter II.5 an algorithm to generate such point sets. However, this algorithm is undesirable in practice as it relies on finding "valid" points through trial and error, similarly to the `Pick_valid` algorithm used

in the locally uniform anisotropic meshes (see Chapter I.4). The heavy cost of this algorithm greatly contributed to the non-practicality of the framework of locally uniform anisotropic meshes.

In practice Fortunately, we obtain good results for 2D domains and surfaces when seed sets are generated through a simple farthest point refinement algorithm. A farthest point refinement algorithm iteratively inserts in the diagram the circumcenter that is farthest from its seed until the prescribed sizing field r_0 is satisfied. Using such refinement algorithm, the dual of our discrete Riemannian Voronoi diagram rapidly becomes an embedded triangulation, even for a relatively small number of vertices (see Section 6.4 and specifically Figure 6.9).

Since we do not enforce power protection for our actual set of seeds, two adjacent Voronoi vertices can be in theory arbitrarily close. This situation also occurs in practice, but while this might seem very problematic, the spreading aspect of our algorithm ensures that only a single diagonal is captured – even if two Voronoi vertices (for the Riemannian Voronoi diagram) are located within the same simplex (see Figure 6.4).

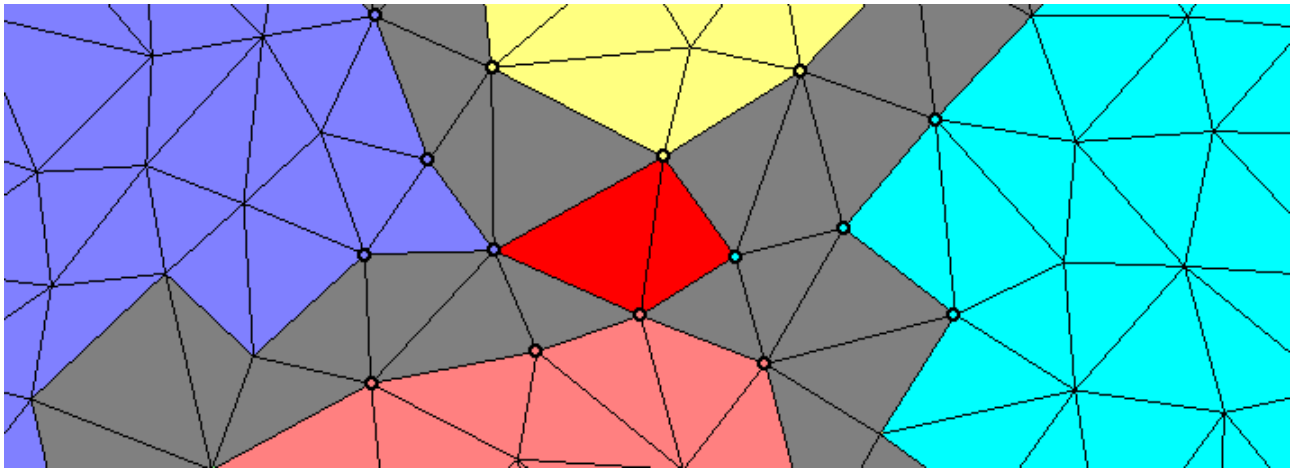

Figure 6.4: As a consequence of not enforcing power protection, Voronoi vertices can be arbitrarily close. The color code is the same as in Figure 6.2. to simplify the comprehension, we have drawn the color at some canvas vertices. This does however not create any issue in the dual triangulation.

The question of the generation of a three-dimensional set of seeds is trickier as a farthest point refinement algorithm is not sufficient to create embedded duals. We will further discuss this issue in Section 6.4.2.

Location of a new seed

Both our theoretical and practical refinement algorithms insert new seed near Voronoi vertices (circumcenters). However, a Voronoi vertex c on the discrete Riemannian Voronoi diagram is actually a canvas simplex $\sigma_{\mathcal{C}}$ whose vertices have all different colors. This is not punctual information and therefore cannot be inserted. Three options are possible to extract a Voronoi vertex from the canvas simplex, ordered below in decreasing amount of simplicity:

- Choose as Voronoi vertex c the canvas vertex of $\sigma_{\mathcal{C}}$ that is farthest from its closest seed.
- Choose as Voronoi vertex c the barycenter of the canvas simplex.

- Recursively break down the canvas simplex $\sigma_{\mathcal{C}}$ into smaller simplices (typically with a barycentric subdivision) and compute the colors of the vertices of these new canvas simplices. Continue until the simplex is under a certain size and pick one of the first two options.

From empirical observations, there is no significant difference between any of the methods and we therefore use the first option since it is the lightest, from a computational point of view.

Insertion of a new seed

A new seed p is easily inserted into an existing discrete diagram: the Voronoi cell of a p is obtained by locating p on the canvas and spreading from the initial canvas simplices with a single front propagation algorithm. The discrete Voronoi cell of p is found when no point reached has a lower distance to p than to its current closest seed.

When the sampling of the seed set is dense on the domain, the insertion of a new seed is a local operation since only a relatively small number of canvas simplices carry the color of any seed.

6.2 Canvas adaptation

The evaluation of geodesic distances is well known to be a computationally expensive (and thus slow) process. To make things worse, the canvas must be composed of a large number of vertices for multiple reasons:

- The canvas needs to be dense with respect to the desired sizing field of the final mesh: the main justification behind the use of geodesics is that the metric field provides significant information that is relevant to the position of vertices and to the combinatorics of the anisotropic triangulation, and which is lost when we only consider the values of the metric field at the seeds.

- The canvas needs to be sufficiently dense with respect to the variations of the metric field and to the reach of the manifold, so that geodesic lengths are accurately estimated.

- The canvas needs to be sufficiently dense so that the nerve is captured.

The requirements mentioned in the second and third items will be precisely described in the theoretical chapters III.7 and III.8.

Remark 6.2.1 *The first item also implies that the metric field must be known in much finer manner compared to the size of the final anisotropic triangulation or the metric field will simply be interpolated between known values, greatly reducing the usefulness of geodesics.*

The combination of the high computational cost of computing geodesic distances and of the required large size of the canvas makes the coloring of the discrete Riemannian Voronoi diagram the (expected) bottleneck of the algorithm. This cost could be reduced in two independent ways: improving the speed of geodesic distance computations and decreasing the amount of geodesic distance estimations. We rely on previous work for the estimation of the geodesic distances and improving the current methods goes beyond the scope of this work. However, we significantly decrease the number of geodesic computations required by changing the nature of the canvas, as we shall now explain.

6.2.1 Anisotropic canvasses

To illustrate the reasoning behind our approach, let us consider a short thought experiment. Assume that a number of consecutive canvas edges form a line in the domain; if the metric were constant along this line, the accuracy of the computation of the geodesic distance would not be improved by a larger number of (smaller) edges. Extrapolating from that simple example, one can intuit that it is not ideal to use an isotropic discretization of the domain as canvas: directions in which the metric field does not vary much could be discretized only sparsely – as long as the canvas honors other constraints (detailed in the bullet points above). In other words, one would rather use an anisotropic canvas. Unfortunately, an anisotropic mesh is exactly what we want to construct.

Salvation comes from the fact that for most geodesic computation methods, a triangulation of the domain is used as canvas but is in fact not necessary: a graph is sufficient. Even better, the embedding of the graph is not required. Both these observations mean that we can use inconsistent star sets obtained from locally uniform anisotropic meshes – a concept introduced and discussed extensively in Chapter I.4 – to construct easily such (anisotropic) graphs. We briefly recall in the next section the framework of locally uniform anisotropic meshes.

6.2.2 Locally uniform anisotropic meshes

The framework of locally uniform anisotropic meshes, introduced by Boissonnat, Wormser, and Yvinec [30], and studied in Chapter I.4, is an attempt at extending the concept of Delaunay triangulation to non-Euclidean settings by computing independently at each vertex its connectivity to other vertices. The algorithm constructs at every vertex $p \in \mathcal{P}$ the Delaunay triangulation in the metric G_p of p, which is the image through the stretching transformation F_p^{-1} of the Euclidean Delaunay triangulation $\mathrm{Del}(F_p(\mathcal{P}))$, with F_p a square root of G_p. The *star* S_p of p in $\mathrm{Del}_p(\mathcal{P})$ is the subcomplex of $\mathrm{Del}_p(\mathcal{P})$ formed by the simplices that are incident to p. The collection of all the stars is called the (anisotropic) star set of $\mathcal{P}$, noted $\mathrm{S}(\mathcal{P})$.

Since the connectivity of each star is set according to the metric at the center of the star, *inconsistencies* may exist amongst the stars of the sites: a simplex σ that appears in $\mathrm{Del}_p(\mathcal{P})$ might not appear in $\mathrm{Del}_q(\mathcal{P})$ with $p, q \in \sigma$. The algorithm refines the set of sites $\mathcal{P}$ until each simplex conforms to a set of a criteria (size, quality, etc.) and there are no more inconsistencies amongst the stars, which is proven to happen.

Unfortunately, the algorithm requires in practice a large number of vertices to solve inconsistencies when dealing with non-uniform metric fields, making it impractical (see Section 4.8 of Chapter I.4). Figure 6.5 shows an example of metric field for which the resolution of inconsistencies is difficult. The algorithm required 4430 vertices to achieve consistency, a number considerably higher than our method, as can be seen on the results produced in Section 6.4.

6.2.3 Anisotropic star set canvasses

Despite the generation of locally uniform anisotropic meshes not being competitive, a graph can be obtained by merging the stars and extracting the 1-dimensional skeleton. Due to the potential presence of inconsistencies in a star set, this graph is not necessarily embedded: the edges of the graph do not necessarily only intersect at endpoints. Nevertheless, it can be used to compute geodesic distances and we are thus able to follow exactly the same algorithm as for the case of an isotropic canvas (Algorithm 16): the seeds are located on the canvas, the vertices of a star set are colored by spreading from the seeds, and the discrete Voronoi diagram is eventually obtained. Although a star set can be created using an isotropic metric field, the advantage of this type of canvasses will come

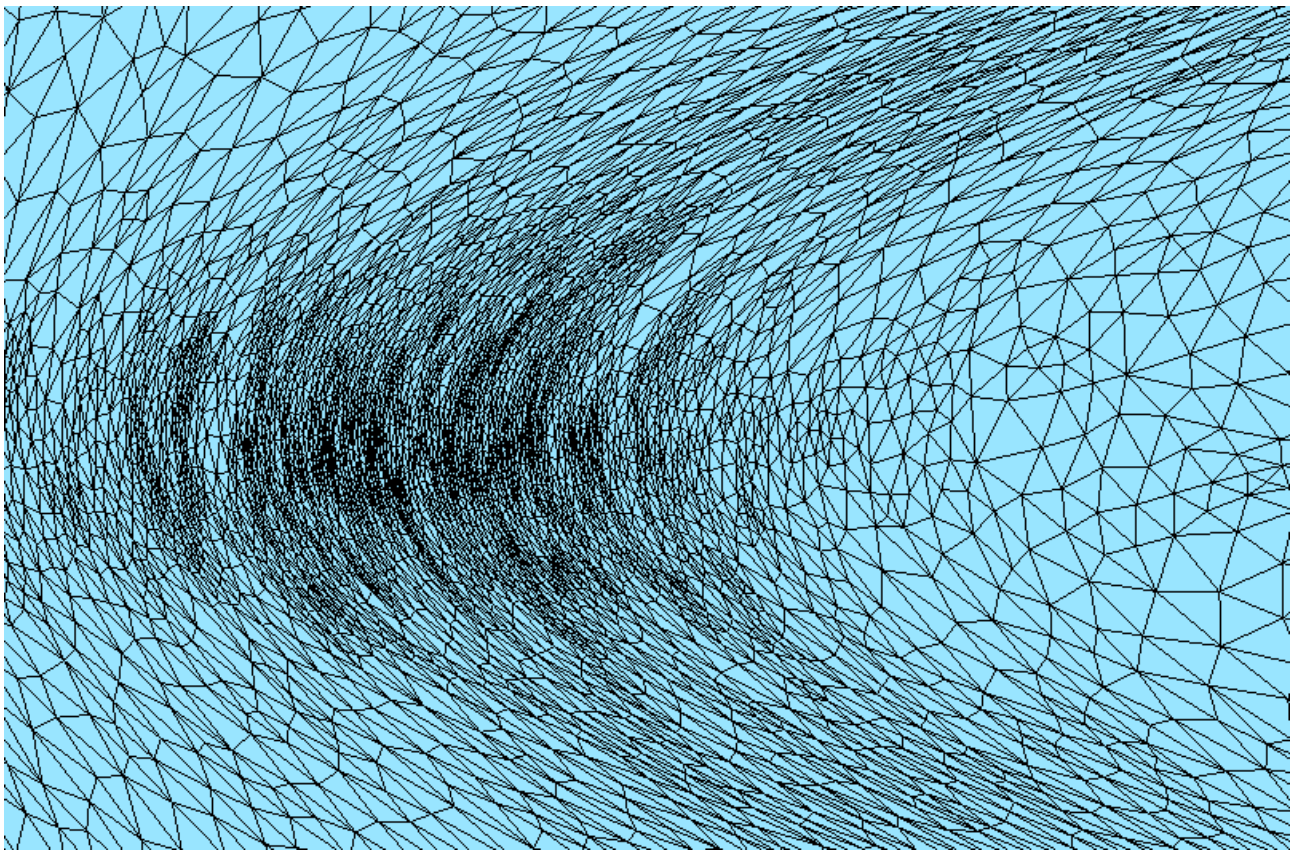

Figure 6.5: Example of a difficult metric field for the local Delaunay algorithm. The algorithm required 4430 vertices to achieve consistency, a number considerably higher than our method as can be seen on the results produced in Section 6.4.

from building a star set which follows the (anisotropic) input metric field. For these reasons, a canvas based on a star set is called an *anisotropic canvas*. The theoretical study presented in Chapters III.7 and III.8 generally assumes that the canvas is an isotropic triangulation for simplicity, but these results can be extended, under some conditions, to anisotropic canvasses.

Remark 6.2.2 *The usage of star-based canvasses is not unlike the stencil-based geodesic distance computation method of Mirebeau [105], which we have described in Section 6.1.1, if one thinks of stars as non-uniform stencils. Although stars offer fewer guarantees than Mirebeau's stencils, they are more flexible and allow one to handle easily the case of (curved) manifolds.*

Orientation of the graph

An anisotropic star in the locally uniform anisotropic mesh framework reflects a local connectivity that is computed according to the metric at the center of the star. An edge that is incident to the center of a star thus possesses a natural orientation: from the center of the star to its other endpoint, on the link. While merging stars, two choices are available:

- Keep the orientation of edges, according to the local connectivities of the stars. One then obtains a directed graph, but this is not a problem in Algorithm 16: spreading from one KNOWN canvas vertex p is only done to its neighbors, the vertices of the link of the star S_p.

- Reciprocate the ownership relations and create an undirected graph by considering that $p \in \mathrm{S}_q$ and $q \in \mathrm{S}_p$ are equivalent. In that case the neighbors of p are the vertices on the link of S_q but also all points q such that $p \in \mathrm{S}_q$. The consequence is a slightly heavier computational cost, but more directions are available from each point, thus improving the accuracy of geodesic distance computations.

The use of reciprocal neighbors (see Section 6.2.3) has in practice no significant impact on the quality of the computation of geodesic distances. This can be explained by observing that we compute geodesics with the algorithm by Campen et al. [36], which uses a vector-valued Dijkstra approach (see Section 6.1.1. With their method, the length of a path over a mesh is estimated by (vectorially) summing edges before measuring the length of the sum. Consequently, the most accurate geodesic distance between two canvas points is not necessarily obtained for the path that is as close as possible to the geodesic between both points. In conclusion, there is no need to maintain an oriented graph.

Nerve of an anisotropic canvas

The extraction of the nerve in the case of anisotropic canvasses is achieved similarly to the case of isotropic canvasses: we traverse the star set and consider the simplices of all the stars. Each simplex is processed by considering the different colors of the vertices of the simplex, and adding the set of corresponding discrete Voronoi cells to the nerve.

Star set canvas generation

Similarly to the isotropic canvas generation, theoretical bounds on the necessary sizing field of the canvas exist, but thankfully do not need to be honored. In practice, the generation of a good star set canvas is even easier than the generation of isotropic canvasses: we simply scale down by a constant factor the sizing field that is chosen for the goal triangulation. Due to the regularity of the point set implied by its the power protected net assumption, this ensures a relatively constant number of canvas edges between two vertices of the final triangulation. The value 0.1 is chosen for this scaling factor and the sizing field of the canvas is thus $0.1r_0$ (with r_0 the sizing field of the set of seeds). Once the refinement process has terminated, we obtain an (anisotropic) star set with Delaunay stars that conforms to the metric field.

As in the case of an isotropic canvas, an issue arises: will the lack of power protection on the point set not create any issue due to Voronoi vertices being too close from one another? This problem is made worse for anisotropic canvasses by the fact that star sets are generally not consistent. Nevertheless, the answer is fortunately here again negative for planar and surface domains and we always obtain embedded duals.

6.2.4 Advantages over isotropic canvasses

To ensure that the nerve of the Riemannian Voronoi diagram is captured in the case of an isotropic triangulation used as canvas, the (uniform) sizing field of the canvas must be small enough such that Voronoi bisectors are clearly distinct (see Section 6.1.6). As the anisotropy ratio increases, Voronoi cells become thinner and the number of canvas vertices required to capture the nerve rapidly grows (Figure 6.6, left and center).

On the other hand, the placement of vertices in an anisotropic canvas is by construction not uniform and does not suffer from the same issue: as the anisotropy of the metric field grows, Voronoi cells and canvas simplices become thinner in tandem. The number of canvas vertices in a Voronoi cell is thus relatively constant regardless of the anisotropy.

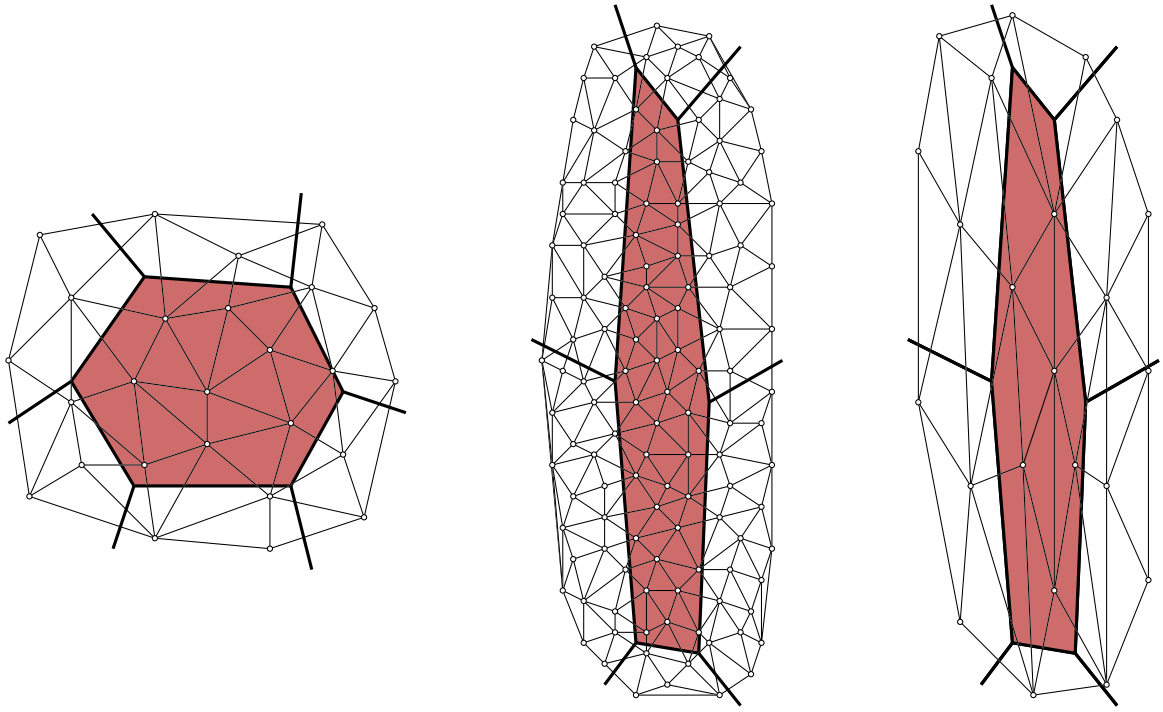

Figure 6.6: Isotropic and anisotropic canvas sampling. In the case of an isotropic canvas, increasing the anisotropy of a cell increases the number of vertices required to properly capture it (left and middle). This is not the case if the canvas can be anisotropic (left and right).

As the star set satisfies a sizing field of $0.1r_0$, the canvas edges are roughly 10 times smaller than the distance between seeds. Consequently, the canvas is composed of approximately 10^n more vertices than seeds, with n the intrinsic dimension of the domain.

The use of an anisotropic canvas greatly decreases the computational time as the number of vertices in the canvas is drastically reduced, without any change in the extracted nerve (see Section 6.4). Figure 6.7 shows an example of the same region of a domain with a triangulation (left) and a star set (right) used as canvasses.

6.3 Implementation

Our implementation is generic: the two types of canvas and the various geodesic distance computation procedures are modules that can be easily swapped in the main algorithm. The CGAL library [2] is used to generate isotropic Delaunay canvases and our implementation (see Section 4.4 in Chapter I.4) of the locally uniform anisotropic mesh framework for the generation of star sets. The algorithm was implemented for planar, surfaces and 3D domains.

6.3.1 Geodesic distance computations

The different geodesic distance computation algorithms described in Section 6.1.1 were implemented and evaluated, with practicality in mind. Although the method proposed by Campen et al. [36] does not offer any theoretical guarantee, its accuracy and convergence were in practice satisfactory: on all the domains considered, increasing the density of the mesh yielded similar geodesic distance maps, meaning that converge is already achieved. The accuracy was estimated by running

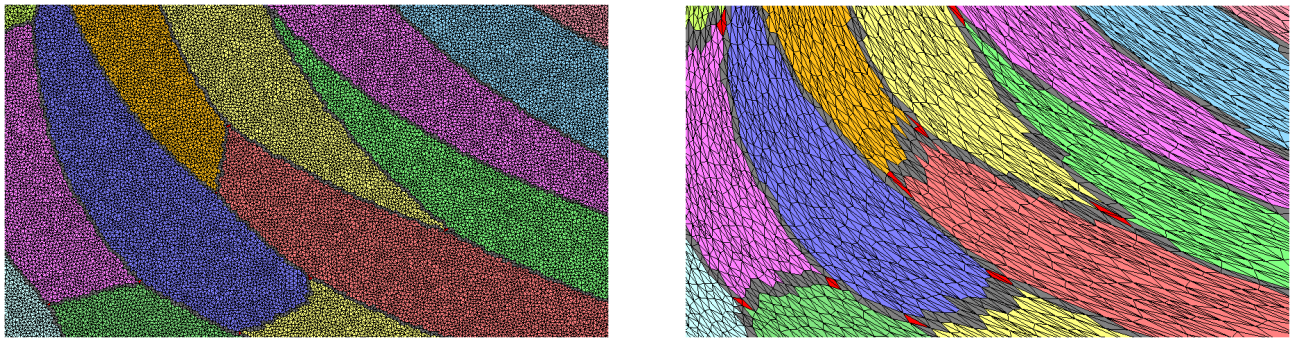

Figure 6.7: Discrete Riemannian Voronoi diagram of the same seed set on two different canvas. On the left, an isotropic triangulation (1M vertices). On the right, an inconsistent star set (40k vertices). The nerves of both discrete Riemannian Voronoi diagrams are identical. The canvas simplices colored in red are witnesses of Voronoi vertices. The canvas simplices colored in grey are witnesses of Voronoi edges.

the method in simple cases where the result is known in advance. Furthermore, the speed of Campen's et al.'s algorithm a significant advantage over other methods. Finally, it requires practically no change to be used in the setting of non-embedded and directed graphs, while still providing accurate results.

6.3.2 Computational speed of the construction of the star set

The computational speed of our implementation of the locally uniform anisotropic mesh framework is low (see Section 4.7 in Chapter I.4), for multiple reasons: the inherent redundancy that comes with a star set (a point is inserted and lives in multiple stars), the necessary combinatorial checks for robustness, and, mainly, the resolution of inconsistencies. Since we do not care about the consistency of our star set here, the meshing algorithm is much faster. Additionally, it is possible to discard all inconsistency-related combinatorial checks (see the maintenance of the refinement queues in Section 4.4.3 in Chapter I.4).

Robustness The algorithm can be accelerated furthermore by trading robustness for speed. Indeed, a number of checks were required in the implementation of the algorithm of Chapter I.4 to guarantee that we are building the correct anisotropic star set. For example, the combinatorial structure of a new star repeatedly uses a filtering structure (a k-d tree) to find nearby points that we might have missed during the construction of the star (see Section 4.4.4 in Chapter I.4). These checks carry a non-negligible computational cost at each iteration but, although they rarely have an impact, are needed if we wish to construct a consistent star set. Since we are now manipulating inconsistent star sets with no care for consistency, we can allow ourselves to not build the exact star set of a set of vertices: a missing vertex never makes a significant difference in the accuracy of the computation of geodesic distances over a large canvas. These tests are thus disabled, and computational speed is increased.

With no particular care needed to be given to inconsistencies, almost all the downsides of the locally uniform anisotropic meshes framework are eliminated: the slow computational time and the large number of vertices were indeed direct consequences of the tedious resolution of inconsistencies.

Parallelization

Parallelization of the locally uniform anisotropic mesh refinement algorithm is difficult (see Section 4.4.6 in Chapter I.4) and was not pursued due to the non practicality of the algorithm.

The results obtained by our approach are much better, renewing the interest in a fast locally uniform anisotropic meshes. Additionally, our method makes uses of geodesic computations, which are notoriously slow, thus making any gain in the algorithm attractive (even if the generation of the canvas is relatively cheap compared to the computation of geodesic distances).

The fact that we can get away with approximate anisotropic star sets (thanks to the vector-Dijkstra method used by Campen et al. [36]) allows us to use a lazy approach to parallelism. Instead of using the local insertion coupled with a lock system, as done for example in the parallel version of the isotropic mesh generator MESH_3 [80] of CGAL, we simply split the domain in zones that are independently meshed. Each zone is made to have a small overlap with the nearby zone, of thickness $0.4r_0$ (meaning an overlap of around two canvas simplices).

Remark 6.3.1 *The partition of the input mesh is computed with the METIS library [84]. Another possibility would be to use the Scotch library [51]. Note that these libraries compute non-overlapping partitions, but it is trivial to enlarge a subdomain with neighboring elements until we are satisfied of its size. In the case of an implicit surface, we first generate a (relatively rough) isotropic mesh of the domain.*

Once a size-abiding star set has been created for each subdomain, we stitch them together by inserting the center of the stars within the overlap region in the stars that are nearby the border in (both regions) region. This is a local and thus swift process.

This approach to parallelism is qualified of lazy because there is no control on the correctness of the stars at the borders and we thus might not building the proper star set. But once again, this is not important here as we only seek a relatively dense and anisotropic star set.

Limitations This subdivision is naturally most efficient when the partition is adapted to our domain. For example, if we are considering a cube of side 10 endowed with the radial shock metric field (see Section A.7 in Appendix A), dividing the cube in 8 smaller identical cubes is not a good partition as most of the anisotropy is only contained in one of the small cubes. This issue was not resolved automatically, but a possible solution would be to use the adapted partitioning algorithm of METIS.

6.3.3 Extraction of the nerve

The extraction of the nerve is performed on the fly during the coloring the canvas: when a canvas simplex is first fully colored, its combinatorial information is extracted and added to the nerve. Since capturing the Voronoi vertices is sufficient to extract all the combinatorial information of the nerve (see Remark 6.1.3), we tag canvas simplices as soon as they have two vertices with identical colors so that their combinatorial information is ignored.

6.3.4 Geodesic paths

To compute curved Riemannian Delaunay triangulations, one must be able to draw geodesic paths and simplices. Geodesic paths are traced by backtracking along the gradient of the geodesic distance map, as described by Yoo et al. [143]. Note that the gradient must be orthogonal to the level sets with respect to the metric field. Figure 6.10 (right) illustrates the gradient map of five Voronoi cells on a discrete Riemannian Voronoi diagram of the "Chair" surface, illustrated on the left.

6.4 Results

In this section, we present results obtained by our discrete Riemannian Voronoi diagram structure and its refinement algorithm. We first exhibit discrete Voronoi diagrams and their corresponding duals for our typical metric fields, before investigating the computational speedup induced by our use of anisotropic canvasses. We compare this new approach to our previous approach, to other definitions of anisotropic Voronoi diagram, and to other anisotropic meshing algorithms. Finally, we expose and discuss issues that arose in the three-dimensional setting.

6.4.1 Duals of the discrete Riemannian Voronoi diagram

As our structure is a close approximation of Riemannian Voronoi diagrams, we can construct two different duals: the curved dual (see Section 2.9 in Chapter 2) and the straight dual whose simplices are defined as the convex hull of the vertices in the domain. We present results for both of these duals, and detail some issues that appeared during the experiments.

We will first consider two-dimensional and surface domains; the case of three-dimensional domains is discussed at the end of the section. The metric fields and domains employed in these results are detailed in Appendices A and B respectively.

Straight Riemannian Delaunay triangulation

By definition, the Riemannian Voronoi diagram captures the metric field more accurately than other methods that typically only consider the metric at the vertices. This additional input of information allows us to construct curved Riemannian Delaunay triangulations, but also has a positive influence on the straight realization of the diagram.

Figure 6.8 shows the different structures involved in our algorithm during the generation of an anisotropic meshes for the sphere endowed with a hyperbolic shock metric field. On the left is the canvas (an isotropic triangulation here); the middle sphere shows the discrete Riemannian Voronoi diagram computed upon this canvas; finally, the right picture shows the dual of the discrete diagram, an anisotropic triangulation. Contrary to the previous approaches introduced and investigated in Chapters I.4 and II.5, no over-refinement is observed, including in the regions where the eigenvectors of the metric field are rotating (where the shock "turns"). The final mesh has slightly fewer than 4000 vertices, which is the number of vertices that was required by our locally uniform anisotropic mesher (Chapter I.4) to generate a mesh of a only small region of that domain (see Figure 6.5).

Figure 6.9 shows the polyhedral surface "Fertility" that we endow with a curvature induced metric field. As for the sphere, the dual rapidly becomes a triangulation during the refinement process: as few as 200 vertices are needed to obtain a triangulation (that is also visually pleasing). Comparatively, our previous algorithms required more than 13000 points to obtain a consistent star set.

Across various experiments, meshes are obtained in a reasonable amount of vertices, often giving close to unit meshes (see Table 6.1).

curved Riemannian Delaunay triangulation

The large amount of additional information that is provided by the canvas allows to construct curved Riemannian Delaunay triangulations, which have not been produced before. Figure 6.10 shows the discrete diagram and the curved Riemannian Delaunay triangulation for the "Chair" surface endowed with a curvature-induced anisotropic metric field. In the computation of the curvature metric field, the value $\epsilon = 0.7$ is used (see Section A.2.1 in Appendix A). The curvature of the

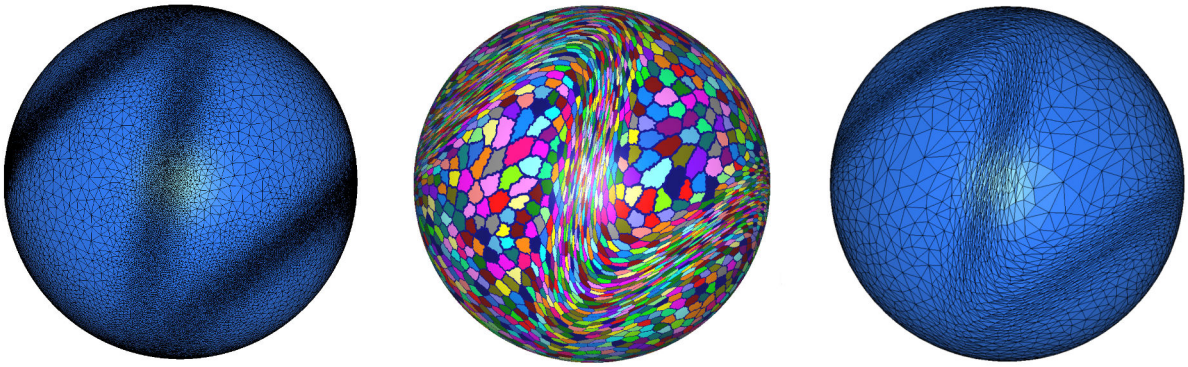

Figure 6.8: The unit sphere endowed with the hyperbolic metric field (approximately 4000 vertices). Isotropic canvas (left), discrete Riemannian Voronoi diagram (center), and straight Riemannian Delaunay triangulation (right).

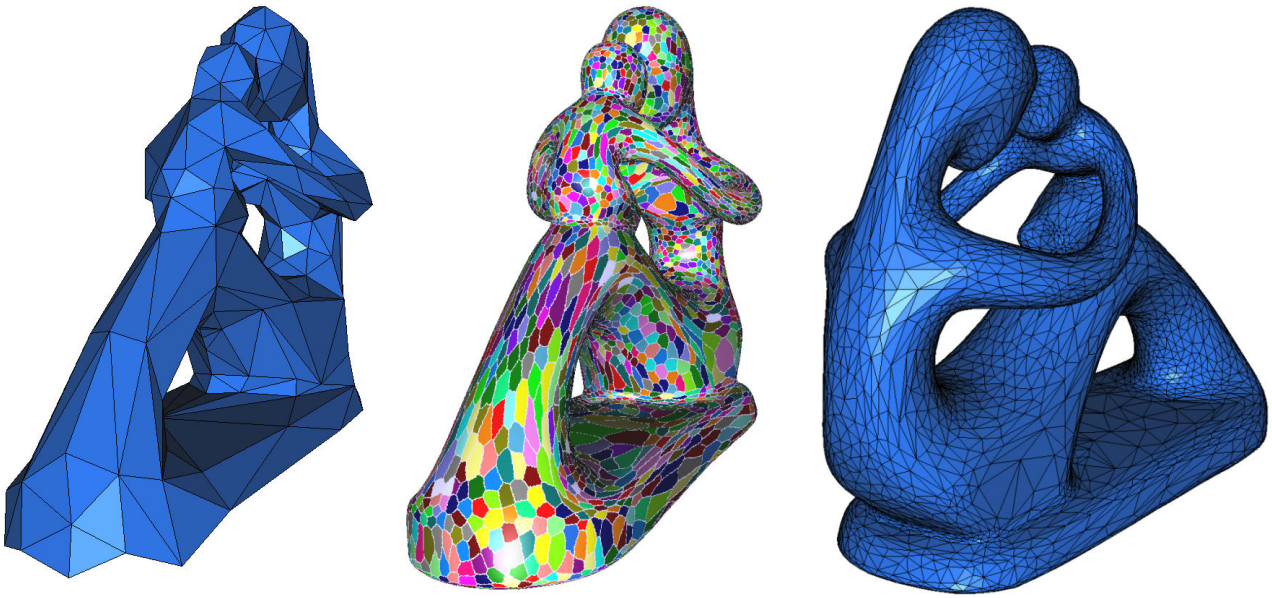

Figure 6.9: the "Fertility" surface endowed with a curvature induced metric field, straight Riemannian Delaunay triangulation with 220 seeds (left), discrete Riemannian Voronoi diagram (center) and straight Riemannian Delaunay triangulation with 6020 seeds (right).

simplices is noticeable and the metric field is well captured with only few curved Riemannian simplices. Comparatively, a straight dual would require more elements to obtain the same approximation

Figure 6.11 shows the curved Riemannian Delaunay triangulation dual of the Riemannian Voronoi diagram for the hyperbolic shock metric field. We obtain an aesthetically pleasing curved mesh that conforms closely to the metric field.

The computation of geodesics is relatively expensive as it requires a gradient descent to draw each geodesic edge. Instead, one can follow the list of ancestors between the two seeds, where an ancestor a is a point of the canvas that provided the minimum of the distance $d(s, a) + d(a, p)$ from a seed s to a point p (see Section 6.1.1 for further details) during the geodesic distance computations.

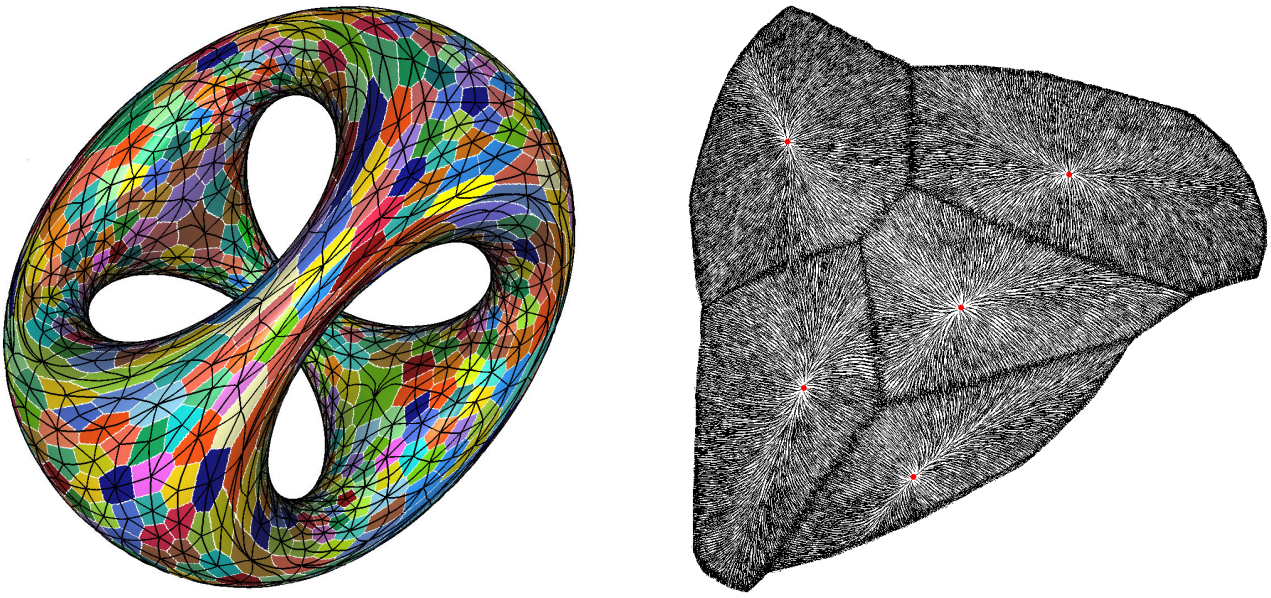

Figure 6.10: On the left, the discrete Riemannian Voronoi diagram of 1020 seeds on the "Chair" surface, with a curvature induced metric field; the edges of the curved Riemannian Delaunay triangulation are traced in black. On the right, the geodesic gradient map used to compute geodesics (for 5 adjacent Voronoi cells picked from the complete diagram). The center of the cells is marked with a red dot and each red line is the gradient at a vertex of the canvas.

Figure 6.12 shows the approximation of the curved Riemannian Delaunay triangulation in Figure 6.11 using segments between ancestors rather than geodesic. Despite only using around 5 segments per geodesic, we obtain a close approximation, at only a fraction of the cost since ancestors can be stored during the process of coloring the canvas.

6.4.2 Three-dimensional setting

The results for three-dimensional domains are more mixed. In the setting of planar and surface domains, it was not necessary for the seed set to be a power protected net as our theory imposes: a farthest point refinement algorithm was enough to create good triangulations. This pleasant result does not extend to the three-dimensional setting: the straight duals of our discrete diagrams generated with a farthest point refinement algorithm are never embedded, due to a low percentage of simplices intersecting nearby elements (see the tetrahedra colored in red in Figure 6.13, right).

Investigating more closely, this is always caused by quasi-cosphericities (the presence of two Voronoi vertices close from one another). The use of power protected nets would entirely prevent such occurrences, and we are thus hopeful that anisotropic meshes could be created this way. Unfortunately, the cost of generating power protected nets in the context of Riemannian Voronoi diagrams is far beyond reasonable and we have thus not progressed farther in this direction.

It should nevertheless be noted that there are fewer issues in three-dimensional triangulations created as duals of discrete Riemannian Voronoi diagrams than in the star set-based approaches, for the same number of vertices. Indeed, the mesh obtained in Figure 6.13 (left) is composed of only 1350 vertices, but is much closer to being a good triangulation than a star set created using the framework of locally uniform anisotropic meshes – which required more than 110000 vertices to create a consistent star set for this same input (see Section 4.7 in Chapter I.4).

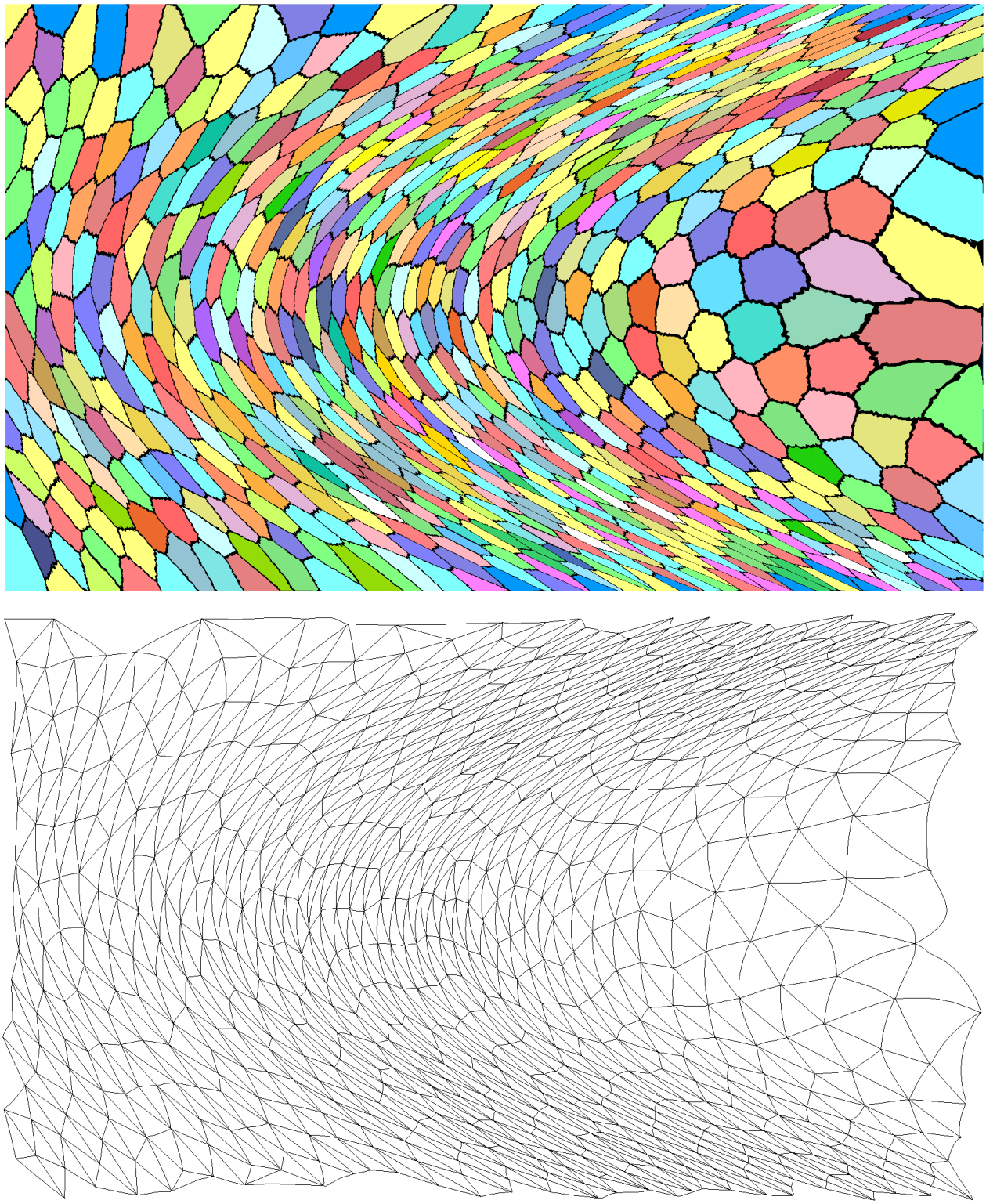

Figure 6.11: Discrete Riemannian Voronoi diagram (top) and curved Riemannian Delaunay triangulation (bottom) of 751 vertices on a planar region endowed with the hyperbolic shock metric field.

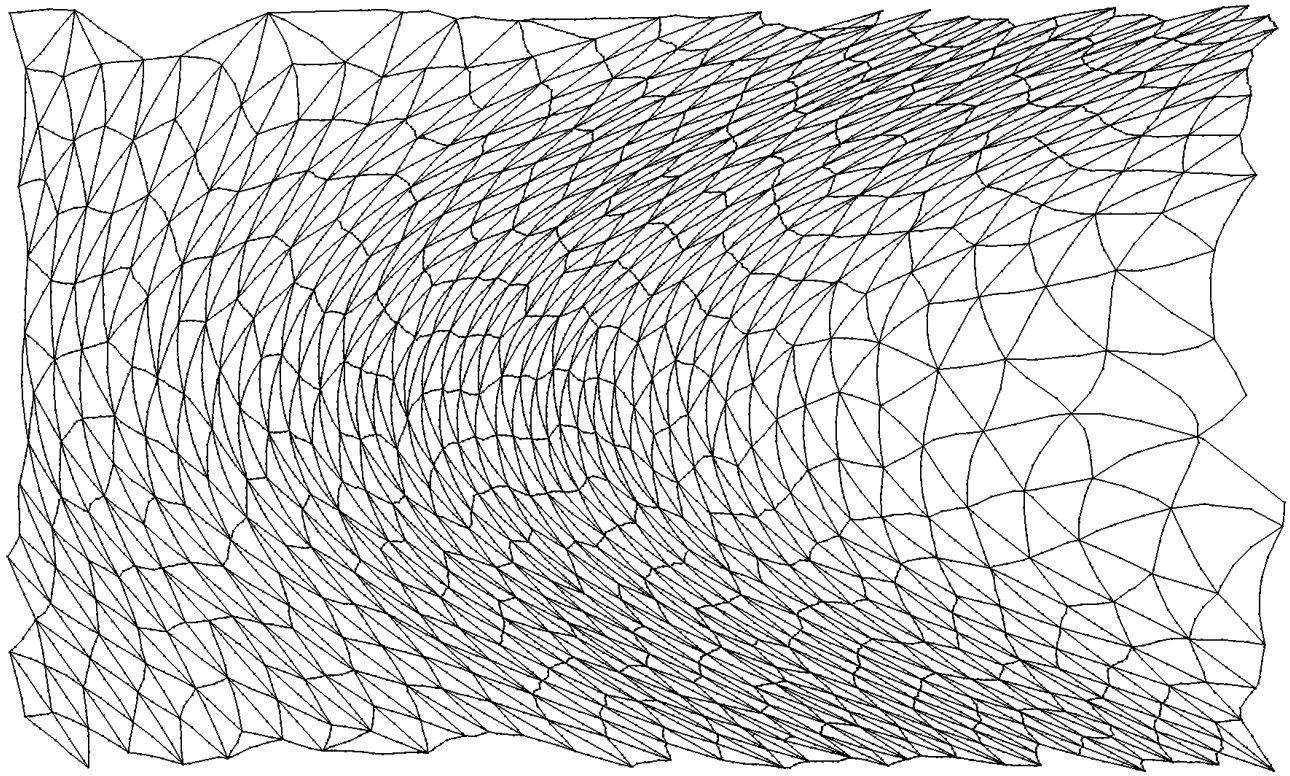

Figure 6.12: Approximating geodesics with segments between ancestors (black dots). Far fewer points are required in comparison with Figure 6.11.

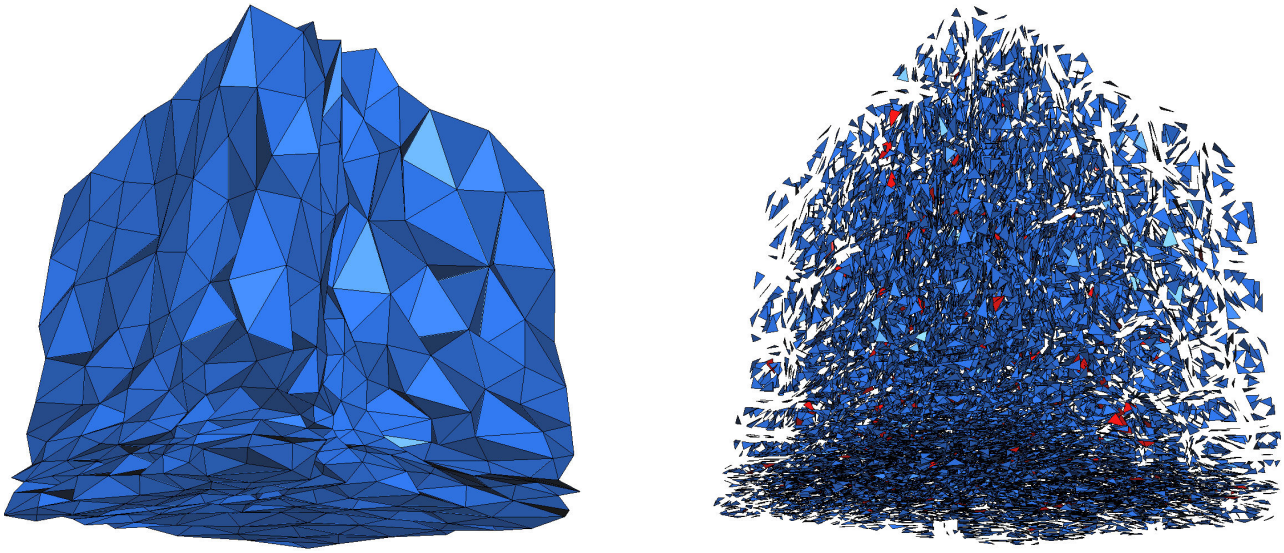

Figure 6.13: Dual of the discrete Riemannian Voronoi diagram for the unidimensional shock metric field. The simplicial complex is composed of 1350 vertices but is not embedded. Problematic elements are drawn in red on the exploded view (faces are scaled down) on the right.

6.4.3 Computational speed

The computational speed of geodesics has always been the justification to approximate Riemannian Voronoi diagrams. We first investigate the computational speed of our algorithm, and specifically at

the gain allowed by the use of a star set canvas. For consistency, the seed set is (exceptionally) assumed to be known beforehand. Note that in all cases, the construction of the nerve and of the geometric realizations of the duals have a negligible cost compared to the computation of geodesic distances.

Figure 6.14 uses an isotropic canvas of a square of side 4 and centered on the origin, endowed with a "Swirl" metric field (see Section A.4 in Appendix A). The canvas is composed of $1,286,862$ vertices and took 29 seconds to generate with MESH_2. The computation of the curved Riemannian Delaunay triangulation and of the straight Delaunay triangulation then took around 40 minutes. Comparatively, the same results were obtained using a star set of only $101,264$ stars (vertices) in around a minute and a half, including 50 seconds to generate the star set.

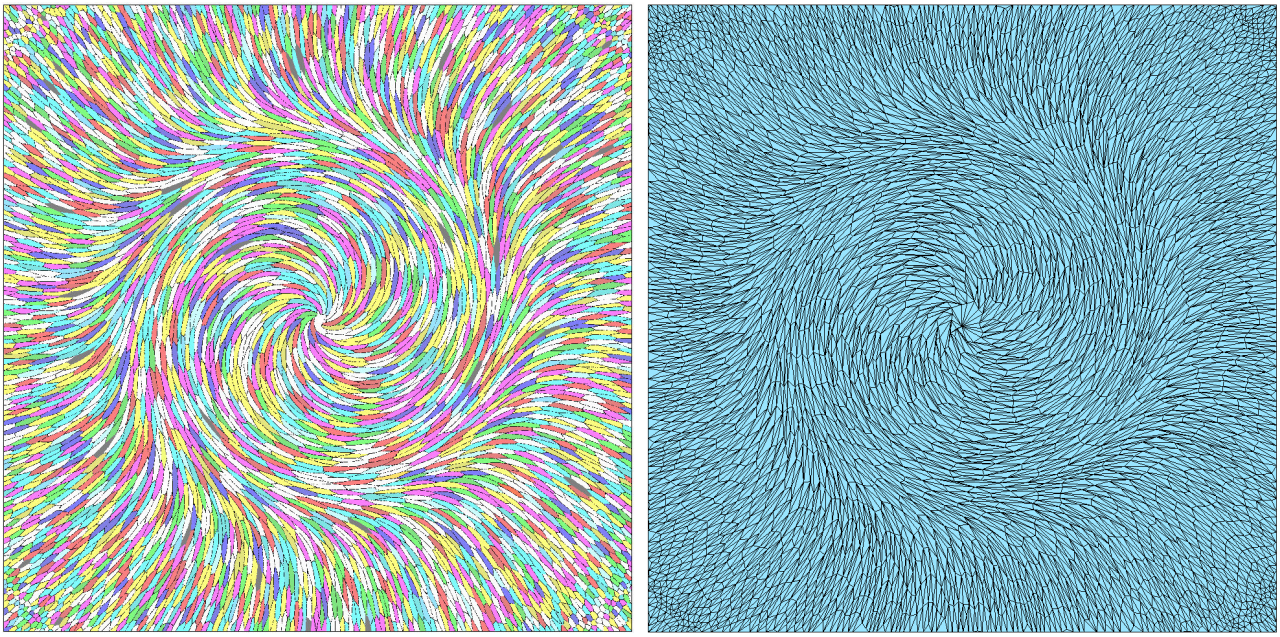

Figure 6.14: On the left, the discrete Riemannian Voronoi diagram of 4000 seeds in a square endowed with a swirly metric field. On the right, the straight dual of the discrete Riemannian Voronoi diagram.

For the same input, increasing the number of seeds to 15000 (see Figure 6.20) yields similar results: the isotropic mesh is generated in 53 seconds and contains 2509300 vertices. Coloring this mesh takes slightly more than an hour. Comparatively, the anisotropic requires 151274 vertices, which were generated in 118 seconds, and the coloring takes less than a minute. This improvement becomes even more significant as the anisotropy increases.

Finally, an isotropic mesh is not even conceivable for larger cases: in our three-dimensional experiments (see Section 6.4.2), we use an anisotropic canvas composed of 1179017 vertices. With our choice of sizing field, the smallest edge has length 0.001, meaning that the isotropic mesh generator would be given the task of meshing a unit cube with an a maximal edge length of 0.0001 (see Section 6.1.6). This would generate a mesh with hundreds of millions of vertices whose coloring would take days.

Generation of seeds

We have so far compared the running times of our algorithms for different canvasses, assuming that the seeds are known.

In reality, we generate the seeds through a farthest point refinement algorithm, as seeds obtained from other means do not produce satisfying Riemannian Voronoi diagram – at least for small seed

sets. This generation of seeds is a major downside of the algorithm, from a computational point of view. Indeed, the insertion of a new seed obliges us to compute its cell, which is done by growing the cell of a new seed from the canvas simplex that contains the seed until it reaches vertices of the canvas that are closer to their seed than to the new seed (see Section 6.1.7). If the seed set is already dense, this is a fast operation as we are only changing the color of a small number of canvas vertices. However, for a certain number of insertions at the beginning (variable depending on the input), large amounts of vertices change colors, which takes a considerable amount of time.

6.4.4 Quality

The quality of a mesh can be evaluated through the measure (in the metric) of its angles and of the lengths of its edges. We evaluate the edge lengths and the angles of the point set shown in Figure 6.19 (the straight Riemannian Delaunay triangulation is shown) for both the curved and straight Riemannian Delaunay triangulations. The angle of a curved Riemannian simplex at a vertex p is given by the angle between the two geodesic edges sent to the tangent space at p using the exponential map (both geodesic edges are then straight edges), which is equivalent to the angle between the tangent of the geodesic edges at a.

The point sets are generated with a sizing criterion intended to create unit meshes, that is curved Riemannian Delaunay triangulations whose geodesic edges have a length close to 1.

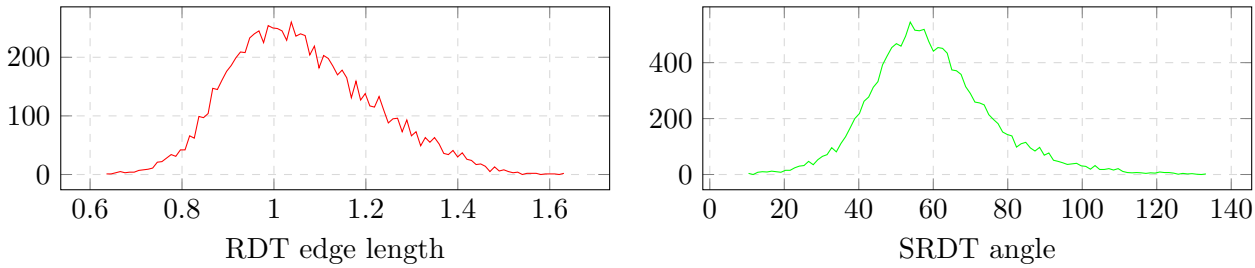

Figure 6.15: Histograms for the optimized mesh shown in Figure 6.19. On the left, edge lengths of the curved Riemannian Delaunay triangulation; on the right, angles values for the straight Riemannian Delaunay triangulation.

Results on the lengths of the edges of the curved Riemannian Delaunay triangulation and on the angles of the straight Riemannian Delaunay triangulation are shown in Figure 6.15 and show good control on both the edge lengths and the angles. We also observe that while the results are naturally better for the curved Riemannian Delaunay triangulation since the point set is generated with respect to the geodesic distance, the quality of the elements of the straight Riemannian Delaunay triangulation stays relatively close to that of the curved Riemannian Delaunay triangulation.

6.4.5 Comparison

Despite great gains brought by the use of anisotropic meshes, the algorithm is still slow. We investigate in this section the benefits of using a geodesic approach by comparing our results with other methods.

Anisotropic Voronoi diagrams

The heavy cost of computing geodesic distances is the fundamental reason behind the approximation of the distance with respect to a metric field with simply a distance with respect to a metric. It is thus particularly interesting to compare our results to anisotropic meshes created for other anisotropic Voronoi diagram, namely using the distance of Labelle and Shewchuk, and the distance of Du and Wang.

We first consider the two-dimensional setting. The hyperbolic shock metric field is a good example of metric field that can be difficult for the anisotropic Voronoi diagrams of Labelle and Shewchuk and Du and Wang. Indeed, other anisotropic Voronoi diagrams who only consider the metric at a single point are prone to inverted elements when the (eigenvectors of the) metric field is rotating, which is the case for the hyperbolic shock metric field. On the contrary, our discrete Riemannian Voronoi diagram takes metric information into account continuously, and is thus not as heavily affected by change in eigenvectors. Consequently, the dual of our diagram becomes a triangulation with fewer seeds than the other anisotropic Voronoi diagram. This is illustrated on Figures 6.16 and 6.17. The number of seeds is fixed for a given example, and the seeds are generated by each algorithm specifically for its distance.

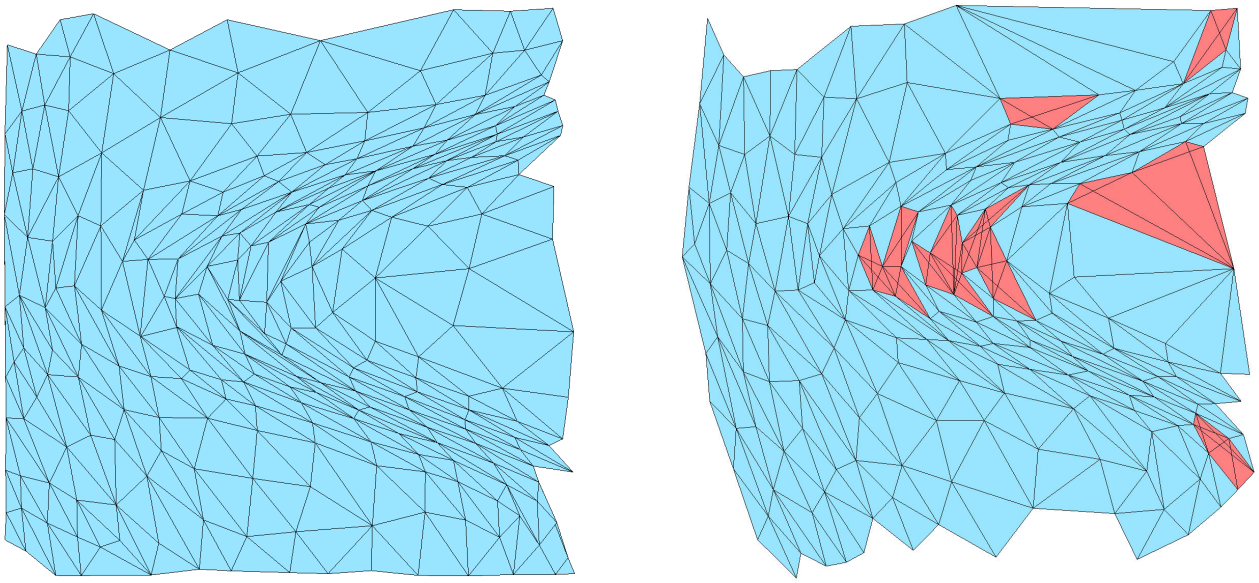

Figure 6.16: Duals of anisotropic Voronoi diagram for a planar domain endowed with the hyperbolic shock metric field (discrete Riemannian Voronoi diagram on the left, AVD of Labelle and Shewchuk on the right). Both Voronoi diagrams contain 250 points by farthest point refinement. Issues preventing the dual from being a triangulation are drawn in red.

Although our diagram does reach the state of having an embedded dual much faster, this advantage is only limited as other anisotropic Voronoi diagrams also reach this state very quickly. The triangulations produced in that short window are also often under-sampled and the result is thus not pleasing visually nor from a quality point of view (for example in Figure 6.16).

Surface Discrete Riemannian Voronoi diagrams produce much better results than other anisotropic Voronoi diagram-based methods on surfaces. Indeed, the geodesic distance captures the geometry of a surface much more accurately and generating an anisotropic triangulation with very few points,

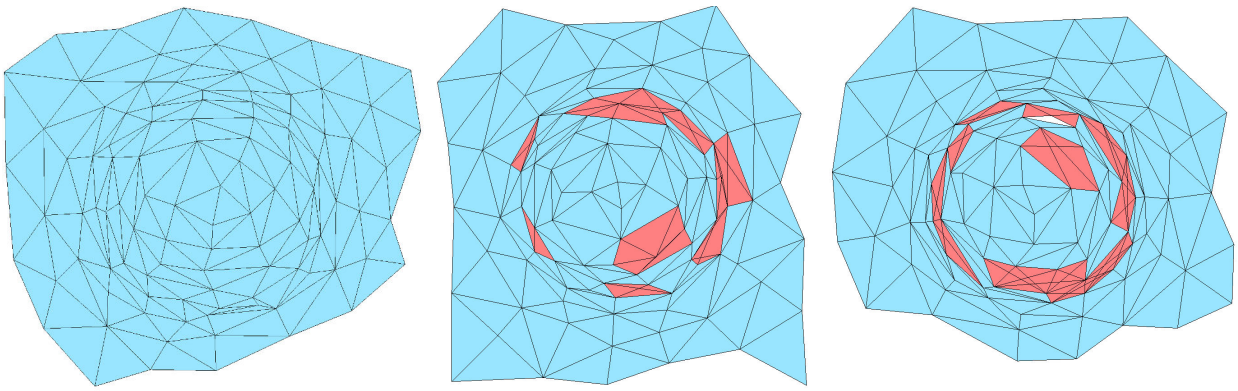

Figure 6.17: Duals of anisotropic Voronoi diagram for a planar domain endowed with a radial shock metric field (discrete Riemannian Voronoi diagram on the left, V^{LS} in the middle and V^{DW} on the right). All Voronoi diagrams contain the same number of seeds. Issues preventing the dual from being a triangulation are drawn in red.

such as the "Fertility" surface composed of 220 vertices in Figure 6.9 (left), is impossible with other anisotropic diagrams.

Star set-based approaches

Two star set-based approaches have been studied previously in this thesis: the framework of locally uniform anisotropic meshes (Chapter I.4) and the use of a modified tangential Delaunay complex to construct the dual of the Labelle and Shewchuk anisotropic Voronoi diagram. In practice, both meshes suffered greatly from inconsistencies, requiring large numbers of vertices to obtain triangulations.

We present below (Table 6.1) statistics on our straight meshes obtained from discrete Riemannian Voronoi diagrams.

Table 6.1: Statistics on the results shown in the previous sections and comparisons with previous results obtained in Table 4.5 (indicated by "LU"). The following metric fields are used: Swirl (Sw), Starred (St), Hyperbolic shock (Hy), Curvature (Cu) (detailed later), Unidimensional shock (UD). Definitions can be found in Appendix A.

	ℓ_{min}	ℓ_{avg}	ℓ_{max}	ψ_{min}	ψ_{avg}	α_{min}	α_{avg}	Q_{min}	Q_{avg}
Square (Sw) - LU	6.1×10^{-5}	0.08	1.38	1	1.19	10.92	41.35	0.27	0.76
Square (Sw)	0.89	1.58	3.25	1	1.25	7.2	44.23	0.15	0.80
Square (Hy) - LU	5×10^{-4}	0.03	0.06	1	1.07	12.56	45.50	0.31	0.80
Square (Hy)	0.64	0.93	1.67	1	1.25	9.78	46.3	0.16	0.82
Fertility (Cu) - LU	0.01	0.50	1.9	1.02	1.82	1.17	39.92	0.03	0.69
Fertility (Cu)	0.18	0.75	6.9	1	1.61	2.17	42	0.02	0.74
Cube 3D (UD) - LU	0.03	0.13	1.01	1	1.08	1.89	43.15	3×10^{-3}	0.73
Cube 3D (UD)	0.32	0.65	1.56	1	1.44	0.1	36.86	2×10^{-3}	0.67

The following measures are listed in Table 6.1:

- **The edge length.** To determine how well the algorithm fares at producing a unit mesh, we compute the lengths of edges with respect to the metric at both endpoints. We list the smallest

(ℓ_{min}), average (ℓ_{avg}) and largest ℓ_{max} lengths of edges.

- **The distortion.** As the resolution of inconsistencies is the criterion with lowest priority and generates the largest amount of vertices, the final distortion of simplices is a good indication of the distortion at which inconsistencies are finally solved. We compute the distortion of all simplices and list the smallest (ψ_{min}) and average (ψ_{avg}) distortion from all the simplices.
- **The smallest angle.** Angles are a good measure of the quality of a triangulation. For each simplex, we compute its smallest angle (dihedral angle in the case of three-dimensional domains). We list the minimum (α_{min}) and average (α_{avg}) of the smallest angles of all simplices.
- **The quality.** A second measure of quality is computed. For planar and surface domains, we use the formula presented by Zhong et al. [144]:

$$Q = 4\sqrt{3}\,\frac{A}{ph}.$$

where A is the area of the triangle, p the perimeter and h the longest edge (all computed in the metric). For cells, we use the quality estimation introduced by Frey and George [71]:

$$Q = 216\sqrt{3}\,\frac{V^2}{A_\Sigma^3}.$$

where V is the volume of the tetrahedron, and A_Σ the sum of the areas of the four facets (all compute in the metric). Both these measures live between 0 and 1, with 1 signaling the highest quality. We list the lowest (Q_{min}) and average (Q_{avg}) quality over all simplices.

For each result, we reproduce the statistics of the corresponding mesh that we had obtained in Chapter I.4 as a locally uniform anisotropic mesh.

The anisotropic triangulations obtained with our new method are much better: thanks to a lower number of vertices required to obtain a triangulation, the edge length is closer to the goal of a unit mesh. A second observation is that despite the visually pleasing results that we obtain with our new method, the angle and the quality of the meshes is lower than the anisotropic meshes produced by Chapter I.4. This is a testament to the rigorousness of the locally uniform anisotropic meshes as every simplex must be a good quality Delaunay simplex in its star for the algorithm to finish. It is however also due to the fact that we have build an anisotropic Voronoi diagram with respect to a geodesic distance but taken the dual with straight simplices, and the position of the seeds might not be optimal for that dual.

Other anisotropic mesh generators

As a first observation, it is clear that our mesher is much slower than most anisotropic mesh generation methods.

The meshes produced by our algorithm do have some benefits compared to other meshing algorithms, for example our planar experiments produce visually pleasing results with decent quality. The most notable advantage is however that the metric field is thoroughly satisfied. Preprocessing the metric field, typically with a Laplacian smoothing, is a technique often used (see for example [73]), which lowers the fidelity to the input metric field and can even nullify the anisotropy, thus resulting in isotropic elements. This is noticeable for example in the "turns" of hyperbolic shock metric field, that is where the eigenvectors of the metrics rotate (pictured for example in Figure 6.5). It is quite common to see meshing algorithms then produce close to isotropic elements in these regions. On the contrary, our algorithm produces a Voronoi diagram that honors the metric field closely and produces triangulations with anisotropic elements that conform much more closely to the metric field.

6.5 Optimization

The optimization of a mesh is often a necessary step in the creation of high-quality meshes. The main tools of mesh optimization are vertex perturbations and combinatorial changes.

Many different optimization methods are known [132], but the concept of centroidal Voronoi tessellations is particularly useful when optimizing structures linked to Voronoi diagrams [64]. Centroidal Voronoi tessellations (CVT) are Voronoi diagrams whose seeds are also the centers of mass (centroids) of their respective cells. The famous Lloyd algorithm [100] iteratively moves the seeds to the center of mass of their respective cell and recomputes the Voronoi diagram of this new seed set. In the Euclidean context, if a Voronoi cell V is partitioned with simplices $\{t_i\}$, the center of mass can be obtained with the formula:

$$c = \frac{\sum_i c_i \, |t_i|}{\sum_i |t_i|}, \tag{6.1}$$

where t_i is the Euclidean area of the simplex t_i.

Many variations have been made to include density fields or using different distances (including anisotropic distances) [62, 63, 64, 98]. Wang et al. [137] used CVTs in the context of isotropic geodesic Voronoi diagrams on surfaces: the center of mass of the geodesic Voronoi cell is computed by sending the Voronoi cell to the tangent space via the exponential map, optimizing in that space, and sending the new centroid back to the surface. Their approach is robust and produces visually pleasing results.

6.5.1 Anisotropic Riemannian CVT

Since our method is based upon the Voronoi diagram, it is natural to try and extend centroidal Voronoi tessellations to our context.

The first idea is to simply use the formulation of Wang et al. [137]: send the Voronoi vertices to the tangent space via with the exponential map, and compute the Euclidean center of mass there. It is however not completely painless as our geodesics are approximated upon a piecewise linear domains, and two different geodesics can sometimes have the same tangent vector at the seed (which cannot happen if geodesics are exactly computed).

While this method gives acceptable results, we introduce a second method, which does not require the exponential map and makes use of the canvas. We approximate the Riemannian Voronoi center of mass of a cell V that corresponds to a seed s with colors c_s with the following formula:

$$c_g = \frac{\sum_i c_i \, |t_i|_{g_i}}{\sum_i |t_i|_{g_i}}, \tag{6.2}$$

where $|t_i|_{g_i}$ is the area of the triangle t_i in the metric g_i and the t_i are canvas simplices whose vertices have color c_s

Remark 6.5.1 *We have also considered including triangles t_i that have at least one vertex of color c_s. This does not change produce any significant change in the output.*

Equation 6.2 is exactly the formula of the Euclidean setting (Equation 6.1), except that the area is computed with respect to the local metric. The canvas conveniently provides the decomposition of a geodesic Voronoi cell in small triangles, making the approximation of c_g accurate.

As its Euclidean counterpart, this algorithm comes with no guarantees (not even for termination) but works well in practice. Figure 6.18 shows the optimization of five seeds in a square centered on the origin and of side 4, endowed with the swirl metric field (see Section A.4 in Chapter A). Three seeds are initially grouped together in the bottom left region.

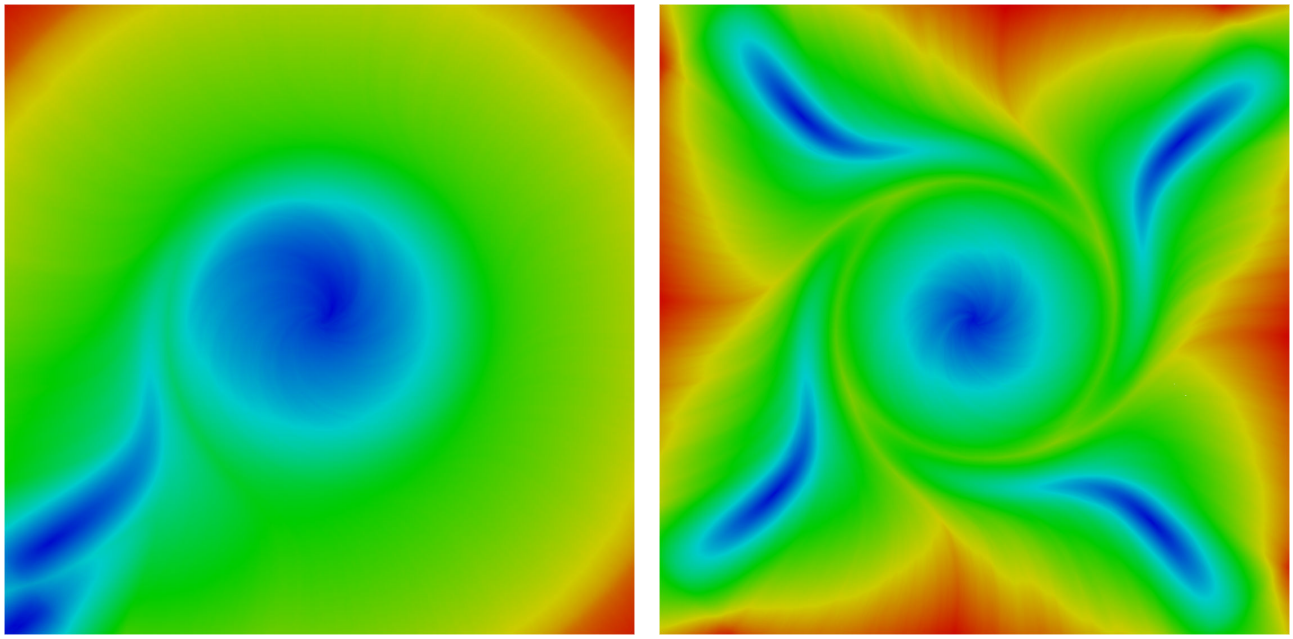

Figure 6.18: Optimization of 5 seeds in a square endowed with a swirl metric field. The canvas is a star set. Colors show the distance from to the closest seed. (Seeds are thus in the middle of the deep blue regions). The algorithm converges quickly and produces a well-spread output.

The formula in equation (6.2) does not extend to surfaces as the result of the weighted sum might not lie on the domain. Projecting the canvas simplices to the tangent planes and computing the center of mass there (or variations of this idea) unfortunately do not yield results as good as for two-dimensional domains. In this setting, we must thus use a process similar to Wang et al. [137], that is to consider the exponential map.

Results

A first observation is that our optimization is functional, despite not relying on the explicit computation of the center of mass through the exponential map. Additionally, there are no noticeable difference in the result of the optimization whether an isotropic canvas or a star set canvas is used (see Figure 6.18).

In Figure 6.19, the initial straight Riemannian Delaunay triangulation of 320 seeds has been optimized with 100 iterations. (Note however that only the first 10 iterations made significant changes to the position of vertices).

With only few elements, the metric field is well captured and aesthetically pleasing. Observe that the elements stay anisotropic in the rotational region, a region that is often smoothed and rendered isotropic by other methods, or is very difficult for the other algorithms introduced in this thesis (see Chapters I.4 and II.5).

Figure 6.20 illustrates a denser version of the mesh shown in Figure 6.14, obtained by reducing the sizing field within the metric field. The final mesh has 15000 vertices and was then optimized with 10 iterations of our optimizer (which was enough to converge).

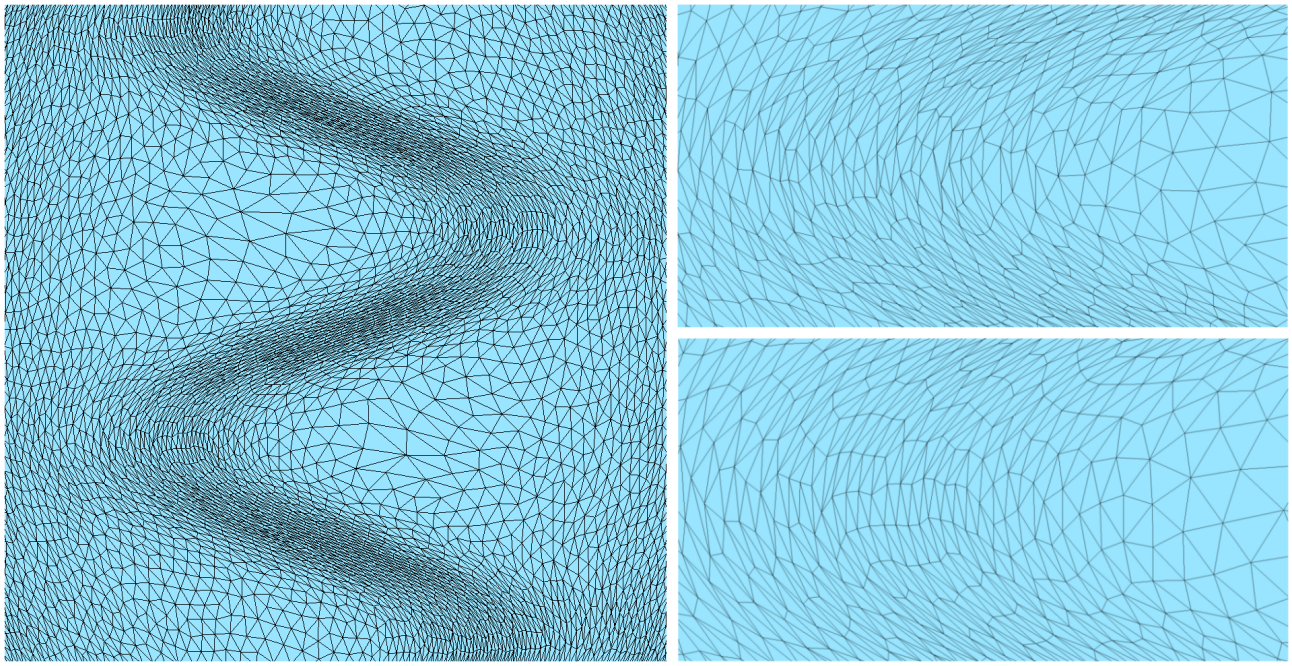

Figure 6.19: The optimized straight Riemannian Delaunay triangulation of 4000 seeds in a planar domain endowed with a hyperbolic shock induced metric field (left). On the right, a zoom on a rotational region of the metric field shows the difference between pre- (above) and post- (bottom) optimization.

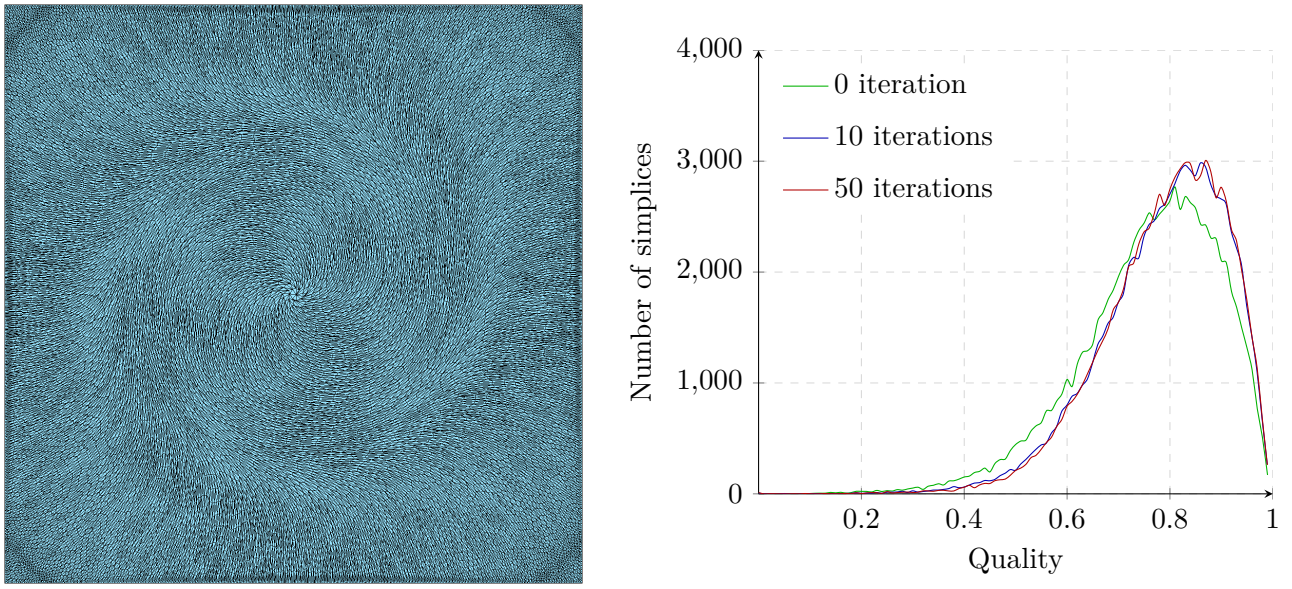

Figure 6.20: Optimization of the a mesh obtained for a square endowed with the "Swirl" metric field. The final mesh has 15000 vertices. On the left, the optimized mesh after 10 iterations. On the right, the quality measure of the facets at three different iterations.

6.6 Conclusion

In this chapter, we have introduced a new approach to create anisotropic meshes based on a discrete structure that approximates the Riemannian Voronoi diagram of a set of seeds. This approach allows

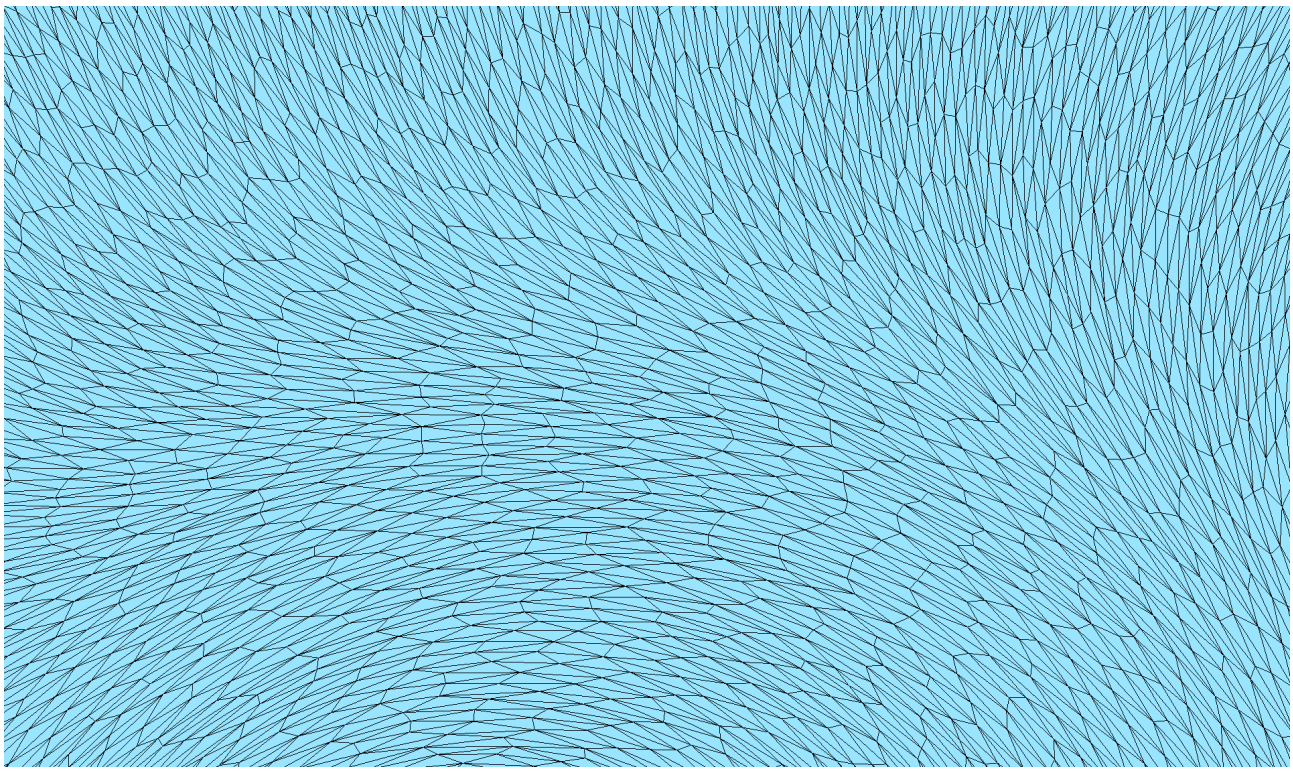

Figure 6.21: Zoom on a region of the optimized swirl surface, after 10 iterations of optimization.

to consider duals composed of straight simplices, but also of curved Riemannian simplices, which has never been considered previously. Our farthest point refinement algorithm generates good results for planar and surface domains and we achieve great improvements over the results we had obtained using the star set-based approaches of locally uniform anisotropic meshes and tangential Delaunay complex (Chapter I.4 and II.5). Finally, we proposed a simple yet effective optimization algorithm that extends the concept of centroid Voronoi tessellations to (anisotropic) Riemannian Voronoi diagrams and produce visually pleasing meshes with good edge and size control. It should also be noted that the points of our diagram are placed for the geodesic distance, and the straight dual is not the most "natural" realization of the nerve.

Comparing our results with other anisotropic Voronoi diagrams, we observe that our diagram requires fewer vertices to possess an embedded dual, notably in regions where the metric field rotates. This advantage only lasts for a relatively small amount of time, and there is – unsurprisingly – no difference between all diagrams as the set of seeds becomes denser.

Locally uniform anisotropic meshes, which produced disappointing results in practice due to over-refinement, provide here a powerful tool to generate anisotropic canvas and allow to greatly reduce the computational cost of computing our diagram by lowering the number of geodesic distance computations. However, the computational time is the obvious limiting factor in our algorithm and the runtime of our algorithm is still large, specifically in 3D, which limits the interest in practical applications. Nevertheless, various improvements are still possible to increase the speed: a natural albeit complicated parallelism can be applied to the coloring of of the canvas, for example.

Our results in 3D are disappointing: the computational time is unreasonable and a farthest point refinement algorithm does not produce embedded duals. Nevertheless, there are fewer issues than for our older methods, and there is good hope that if we could generate power protected nets in reasonable

time, we would obtain good results. Additionally, there is little doubt that this type of seed sets could be used by other anisotropic Voronoi diagrams to create good triangulations: we have even already proved that under certain conditions a power protected net would produce a diagram whose dual is embedded. However, the generation of protected point nets is currently tedious and this is thus not an option.

We shall show in Chapters III.7 and III.8 that under some conditions on the quality of the point set and on the density of the canvas, we can prove that our structure captures the structure of the Riemannian Voronoi diagram and that our duals – both Riemannian and straight – are embedded.

7. Theoretical study of 2-dimensional discrete Riemannian Voronoi diagrams

The discrete Riemannian Voronoi diagram, introduced in Chapter III.6, is a structure that aims to build an accurate approximation of Riemannian Voronoi diagrams of manifolds endowed with an arbitrary metric field. As its name indicates, we consider discrete information by coloring the vertices of an underlying triangulation called the canvas and extracting the combinatorial information from the simplices of the canvas to build the dual of the diagram.

Our interest residing in anisotropic triangulations, we have investigated in Chapter III.6 two possible duals of our diagram, using either curved or "straight" Riemannian simplices. With the former, we obtain the curved Riemannian Delaunay complex, more intuitive since a discrete Riemannian Voronoi diagram is defined using the geodesic distance but also a mostly theoretical realization. With the latter, we obtain a more classical complex that is both easier to construct and to use. For either realization, these complexes are not necessarily triangulations, in the sense that they are not necessarily embedded. We investigate in this chapter the theoretical aspect of our structure and specifically under which conditions the duals can be proven to be embedded.

Curved Riemannian Delaunay triangulations have previously received some theoretical attention: Leibon studied the curved dual of the Riemannian Voronoi diagram in 2D [90] and gave conditions such that the curved Riemannian Delaunay complex is a triangulation. Together with Letscher [91], he extended this study to curved Riemannian Delaunay triangulations of any dimension, but their analysis was shown to be flawed and corrected by Boissonnat et al. [17], at the expense of additional conditions. Dyer et al. [66] improved the two-dimensional and surface work of Leibon, with better bounds.

In this chapter, we restrain ourselves to the setting of two-dimensional manifolds. This allows us to present proofs that are simpler and to obtain slightly better bounds than in the next chapter, which will extend results to any dimension. Results on Euclidean Voronoi diagrams and Riemannian geometry presented in Chapter 3 provide us the tools necessary to prove the theoretical soundness of our approach.

Contributions We prove the theoretical soundness of our discrete Riemannian Voronoi diagram structure (introduced in Chapter III.6) in the settings of a 2-manifold embedded in either $\mathbb{R}^2$ (planar domain) or $\mathbb{R}^3$.

We exhibit sampling conditions such that the discrete and Riemannian Voronoi diagrams have combinatorially equivalent nerve, such that the dual of the Riemannian Voronoi diagram is a triangulation and, finally, such that the geodesic edges of a curved Riemannian Delaunay triangulation can be "straightened" without losing the embeddability of the dual, thus obtaining the embeddability of the straight dual of our diagram.

While results on the embeddability of the dual of a curved Riemannian Voronoi diagram in a two-dimensional setting are already known, we offer a new point of view by using an approach based upon Whitney's lemma rather than Edelsbrunner and Shah's topological ball property [68] (as was done by Dyer et al. [66]). Along the way, we fix a small flaw in Dyer et al. [66] in a result on the

intersection of convex bodies.

Contents

7.1 Background

We first recall some concepts introduced in the previous chapter which will be useful in this theoretical work. Details can be found in Chapter III.6.

We consider a set of seeds $\mathcal{P}$ in a domain Ω, and each seed is attributed a color. The domain is endowed with an arbitrary metric field g, and the geodesic distance between two points, denoted by d_g, is the length of the shortest path between these two points. The Voronoi diagram constructed with the geodesic distance is the Riemannian Voronoi diagram $\mathrm{Vor}_g(\mathcal{P})$. As this diagram is difficult to compute, we approximate the geodesic distance and the diagram $\mathrm{Vor}_g(\mathcal{P})$ by considering a discrete underlying structure, the canvas $\mathcal{C}$, whose vertices are colored according to their closest seed. This coloring is obtained through a multiple-front Dijkstra algorithm from the seeds. The discrete Voronoi cell of a seed is the set of simplices that possess at least one vertex colored with the color of the seed. The discrete Riemannian Voronoi diagram $\mathrm{Vor}_g^d(\mathcal{P})$ is the set of discrete Voronoi cells. An example of such a diagram is illustrated on Figure 7.1. The nerve of a Voronoi diagram $\mathrm{Vor}_d(\mathcal{P})$ is a formal way to express its combinatorial structure and is defined by

$$\mathcal{N}(\mathrm{Vor}_d(\mathcal{P})) = \{\{\mathrm{V}_d(p_i)\} \mid \cap_i \mathrm{V}_d(p_i) \neq \emptyset, p_i \in \mathcal{P}\}.$$

The straight and curved duals of a Voronoi diagram are realizations of the nerve with vertices in $\mathcal{P}$ and using respectively straight and curved Riemannian Delaunay simplices (see Sections 2.7 and 2.8 in Chapter 2 for additional details on the definition). If a dual is embedded in the domain, we say that it is a triangulation.

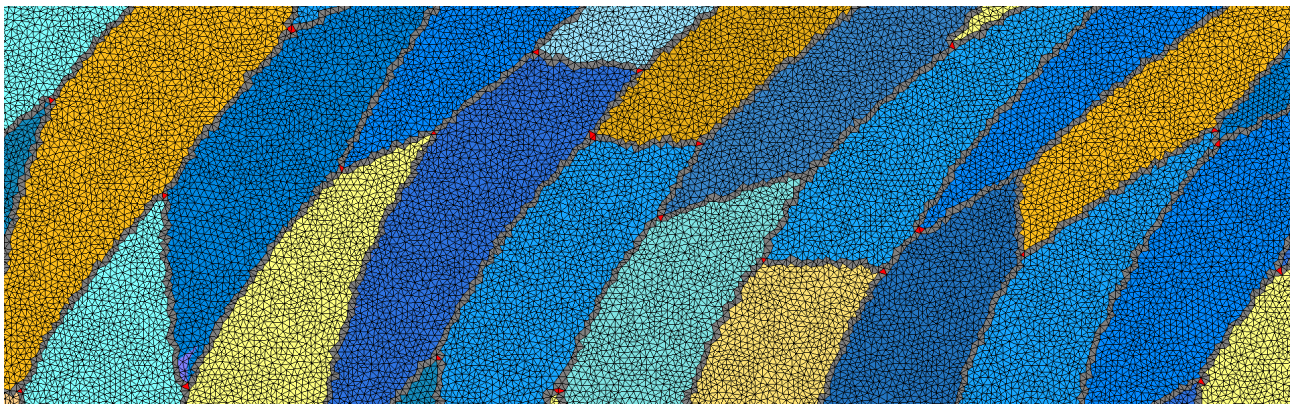

Figure 7.1: Discrete Riemannian Voronoi diagram over an isotropic canvas.

7.2 Overview

Contrary to the Euclidean setting, the dual of a Riemannian Voronoi diagram is generally not a triangulation, even if the points are in general position. To prove the embedding of both the curved and straight duals, one might first think of working directly on the discrete Riemannian Voronoi diagram and exposing sufficient conditions, similarly to the work of Cao et al. [39] on digital Euclidean 2D Voronoi diagrams. Although Cao et al. are able to provide conditions such that the dual of their digital Voronoi is a triangulation, the case of a Riemannian manifold endowed with a metric field is much more complicated.

Instead, we follow a less direct path. We first give requirements such that the discrete and Riemannian Voronoi diagrams have the same nerve (combinatorial structure). This result is then combined with known results on curved Riemannian Delaunay triangulations [17, 90, 65] – which we reformulate with a new approach – to provide sampling requirements for the curved dual to be a triangulation. Finally, we prove that under some sampling conditions, "straightening" geodesic edges will not cause the loss of the embedded property of the realization. The fact that the curved dual is a triangulation will therefore be both a result and a step in the proof that the straight dual is a triangulation.

We first give an overview of the three parts of this chapter.

7.2.1 Equivalence of the nerves of the discrete Riemannian Voronoi diagram and the Riemannian Voronoi diagram

The first part of the theory aims to show that while our discrete Riemannian Voronoi diagram uses by definition only discrete information, its combinatorial structure can be under sufficient conditions identical to the one of the Riemannian Voronoi diagram. When this is the case, we say that the nerves of the diagrams are *equivalent* and write $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}_g(\mathcal{P}))$.

Remark 7.2.1 *This equivalence is only combinatorial: the nerve of a Voronoi diagram being a set of sets of Voronoi cells, it is impossible for both nerves to be equal since the Voronoi cells of the same seed is different in each diagram. However, there is a natural equivalence between a set* $\{\mathrm{V}_g(p_i)\}$ *and a set* $\{\mathrm{V}_g^d(p_i)\}$ *(where* $\mathrm{V}_g^d(p_i)$ *is the cell of* p_i *in the discrete Voronoi diagram, see Definition 6.1.2 in Chapter III.6) of the same seeds. For this reason, we use the* $\cong$ *notation.*

As noted in Remark 6.1.3 in Chapter III.6, the combinatorial structure of the discrete Riemannian Voronoi diagram can be extracted by only looking at the maximal canvas simplices whose vertices

have all different colors, as long as the dual complex is pure (but we shall not consider the degenerate case where it is not). To have equivalence between the nerves of the diagrams, we must ensure that

(1) for every Voronoi vertex in the Riemannian Voronoi diagram, there is at least one canvas simplex that possesses the corresponding colors;

(2) no canvas simplex witnesses combinatorial information that does not belong to the nerve (that is, no canvas simplex has vertices whose colors are those of non-adjacent Riemannian Voronoi cells).

Requirements on the canvas

To satisfy Condition (1), the density of the canvas must depend on the smallest distance between Voronoi vertices to ensure the equivalence. Figure 7.2 illustrates this observation: on the left, the canvas is too sparse to capture accurately $\mathcal{N}(\mathrm{Vor}_{\mathbb{E}}(\mathcal{P}))$: the canvas simplices marked in red will create simplices that do not belong to the dual of the (exact) Riemannian Voronoi diagram; on the right, the canvas is sufficiently dense and the green simplices correctly capture the combinatorial information of the Riemannian Voronoi diagram. To satisfy Condition (2), it is sufficient to prevent any canvas

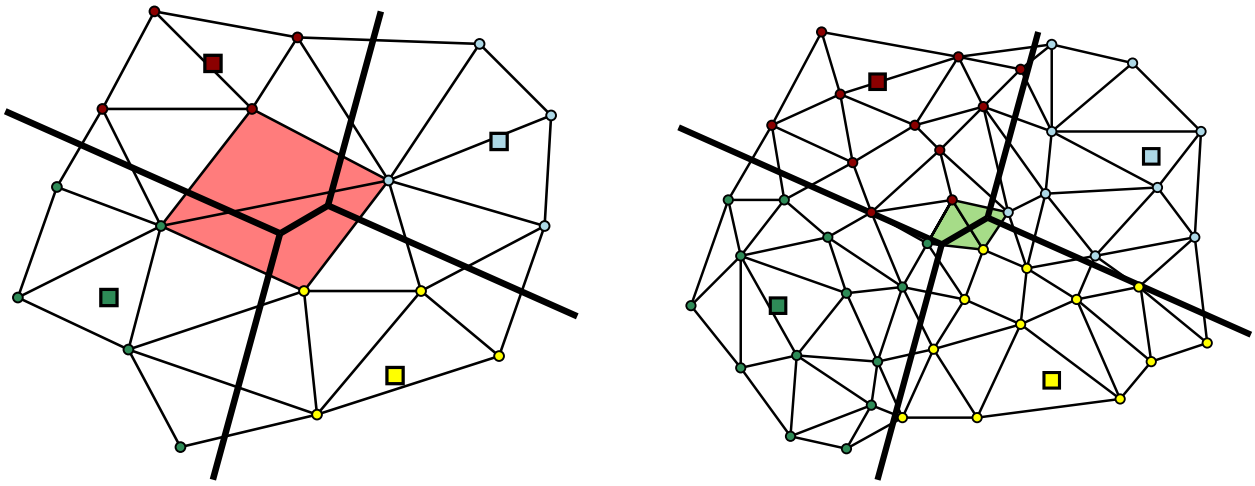

Figure 7.2: Two discrete Riemannian Voronoi diagram in the simple case of a plane endowed with the Euclidean metric field. The seeds are drawn with squares, the canvas with thin black segments and the canvas vertices with circles. The Riemannian Voronoi diagram is drawn with thick black segments.

edge from having endpoints in two non-adjacent Voronoi edges. Both conditions are thus satisfied, provided that the canvas is sufficiently dense.

Requirements on the set of seeds

Using seeds generated by other methods yields in practice unsatisfactory results. Consequently, the canvas is also used to generate seeds (see Section 6.1.6 in Chapter III.6). Since the canvas is generated before the seeds, seeds cannot be – in theory – in arbitrary position, otherwise the distance between Voronoi vertices could be arbitrarily small and it would be impossible to preemptively construct a canvas that reliably captures the nerve. The concept of power protected nets, recalled in Sections 2.12.1 and 2.12.2 of Chapter 2, will provide these guarantees for the seed set as we have shown in Chapter 3 that a Voronoi diagram built from a power protected set of seeds has separated adjacent Voronoi vertices (see specifically Lemma 3.2.4).

Approach

We obtain the equivalence of the nerves incrementally, observing that a manifold with an arbitrary metric field is locally well approximated by a flat domain with a uniform metric field, which is only a linear transformation away from considering the Euclidean metric field. In this simpler setting, sufficient conditions to obtain the equivalence of the nerves are exposed (Theorem 7.3.1) using the separation of the Voronoi vertices induced by the power protected net hypothesis. These conditions are finally pulled back to the general setting, and the equivalence of the nerves for an arbitrary 2-manifold endowed with an arbitrary Lipschitz-continuous metric field is shown in Section 7.3.3.

7.2.2 Embeddability of the curved Riemannian Delaunay triangulations

Once it is proven that our discrete diagram has the same combinatorial structure as the Riemannian Voronoi diagram, we can simply build upon known results that have been introduced on the Riemannian Voronoi diagram and the curved Riemannian Delaunay triangulation. We show that, under sufficient conditions, the dual of the Riemannian Voronoi diagram – and thus of our discrete diagram, assuming that sufficient conditions are satisfied – can be embedded as a triangulation (mesh) of the point set, with edges drawn as geodesic arcs (Section 7.4). Although the results we use on the embeddability of the curved Riemannian Delaunay triangulation are known, our approach and proofs are based on Whitney's lemma instead of Edelsbrunner and Shah's closed ball property [69]. The main result is given in Theorem 7.4.18.

7.2.3 Embeddability of the straight Riemannian Delaunay triangulations

The last part of the chapter is devoted to exposing requirements such that the straight dual of a Riemannian Voronoi diagram is a triangulation. We assume that conditions are met for the curved Riemannian Delaunay triangulation to be embedded and give conditions such that "straightening" the curved simplices into straight simplices does not produce any inversion.

The key intermediary result is the computation of a bound on the distance between a point on a geodesic edge and its equivalent on the straightened edge. This bound depends on the quality of the seed set and on the local distortion of the metric field and is computed in Section 7.5. With the knowledge of this bound, we encompass the geodesic edges within "tubes" (in beige in Figure 7.3). These tubes can be shown to not cover simplices entirely, thus proving that no inversion appear when deforming the curved Riemannian Delaunay triangulation into a straight Riemannian Delaunay triangulation (Theorem 7.5.3).

7.2.4 Nature of the canvas

To simplify computations, we assume in this chapter that the canvas is an isotropic Delaunay triangulation of the domain. We further assume that this triangulation is generated through a Delaunay refinement algorithm driven by a uniform sizing field $h_{\mathcal{C}}$ that bounds the circumradius of Delaunay simplices. This assumption does not reduce the genericity of the approach: these proofs can be easily extended to star set canvasses as a star set canvas eventually becomes a consistent triangulation using well-chosen canvas vertices (see Chapter I.4).

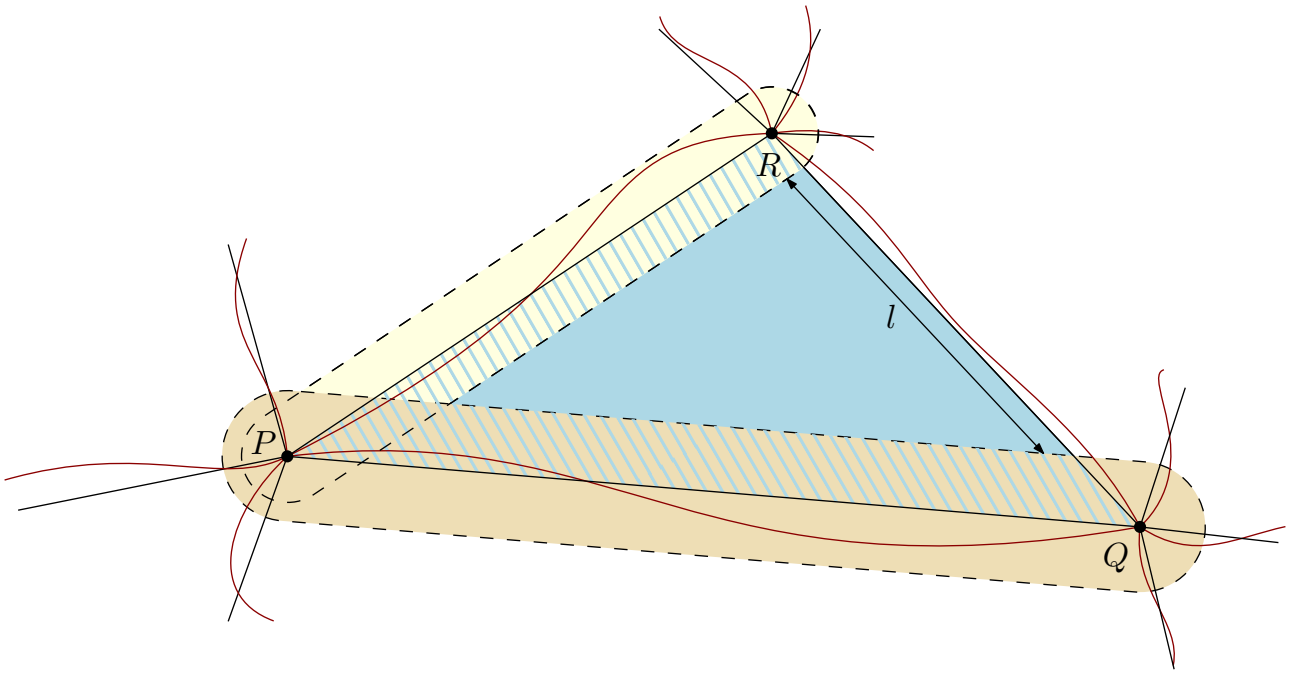

Figure 7.3: The geodesic edges between seeds P-R and P-Q lie in their respective protection tubes (colored in beige) around the straight edges

7.3 Equivalence of the nerves of the discrete and Riemannian Voronoi diagrams

The first step on the path to proving that the dual of our discrete Riemannian Voronoi diagram is a triangulation is to show that the discrete and Riemannian Voronoi diagrams have combinatorially equivalent nerve. Our analysis is divided in four parts.

- We first consider at the most basic case of a flat domain endowed with the Euclidean metric field. Results from Chapter 3 on the shape of Euclidean Voronoi cells and the separation of Voronoi vertices allow conditions to be deduced on $\mathcal{P}$ and $\mathcal{C}$ such that the nerves are equivalent.
- The requirements exposed in the basic setting can be transported to the setting of a uniform metric field through the use of a stretching transformation computed from the metric.
- An arbitrary Riemannian metric field is considered over the plane, and the fact that a metric field can be locally well approximated by a uniform metric field is used to extend the results from the previous cases.
- We solve the case of a general surface by locally approximating the manifold at a point with its tangent plane.

We start with the setting of a flat domain endowed with the Euclidean metric field in the next section.

7.3.1 Euclidean metric field

The setting of a planar domain endowed with the Euclidean metric field is the simplest setting possible. As explained in Section 7.2, we must ensure that Voronoi vertices cannot be arbitrarily

close to each other, otherwise we cannot guarantee that canvas simplices capture the correct nerve. We show that the notions of net and power protection can be employed for the seed set and give conditions on the density of the canvas such that the nerves of the diagrams built from these points sets are equivalent.

The main result is given by Theorem 7.3.1.

Theorem 7.3.1 *Assume that $\mathcal{P}$ is a δ-power protected (ε, μ)-net of $\mathbb{R}^2$ with respect to $g_{\mathbb{E}}$. Denote by $\mathcal{C}$ the canvas, a Delaunay triangulation of $\mathbb{R}^2$, with sizing field $h_{\mathcal{C}}$. If*

$$h_{\mathcal{C}} < \frac{\delta^2 \mu}{8\varepsilon^2} \sqrt{1 - \left(\frac{\mu}{2\varepsilon}\right)^2},$$

then

$$\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}_g(\mathcal{P}))$$

.

Lemmas 7.3.2 and 7.3.3 are intermediary results used to prove Theorem 7.3.1, which correspond nicely to Conditions (1) and (2).

Lemma 7.3.2 *Assume that $\mathcal{P}$ is a δ-power protected (ε, μ)-net of $\mathbb{R}^2$ with respect to the Euclidean metric field $g_{\mathbb{E}}$. Consider a seed $p_0 \in \mathcal{P}$ and let $\mathrm{V}_{\mathbb{E}}(p_0)$ be its corresponding Voronoi cell in $\mathrm{Vor}_{\mathbb{E}}(\mathcal{P})$. If*

$$h_{\mathcal{C}} < \frac{\delta^2 \mu}{8\varepsilon^2} \sqrt{1 - \left(\frac{\mu}{2\varepsilon}\right)^2},$$

then no edge of $\mathcal{C}$ intersects two non-adjacent Voronoi faces of $\mathrm{V}_{\mathbb{E}}(p_0)$.

Lemma 7.3.3 *Assume that $\mathcal{P}$ is a δ-power protected (ε, μ)-net of $\mathbb{R}^2$ with respect to the Euclidean metric field $g_{\mathbb{E}}$. Consider a seed $p_0 \in \mathcal{P}$ and let $\mathrm{V}_{\mathbb{E}}(p_0)$ be its corresponding Voronoi cell in $\mathrm{Vor}_{\mathbb{E}}(\mathcal{P})$. If*

$$h_{\mathcal{C}} < \frac{\delta^2 \mu}{8\varepsilon^2} \sqrt{1 - \left(\frac{\mu}{2\varepsilon}\right)^2},$$

then every Voronoi vertex of $\mathrm{V}_{\mathbb{E}}(p_0)$ is witnessed by a canvas simplex.

Observe that, Lemma 7.3.2, by preventing canvas simplices from capturing combinatorial information that is not in $\mathcal{N}(\mathrm{Vor}_{\mathbb{E}}(\mathcal{P}))$, ensures that for every element (a set of Voronoi cells) in $\mathcal{N}(\mathrm{Vor}_{\mathbb{E}}^d(\mathcal{P}))$ there exists a combinatorial equivalent in $\mathcal{N}(\mathrm{Vor}_{\mathbb{E}}(\mathcal{P}))$. Conversely, Lemma 7.3.3 ensures that for every element in $\mathcal{N}(\mathrm{Vor}_{\mathbb{E}}(\mathcal{P}))$, there exists a combinatorial equivalent in $\mathcal{N}(\mathrm{Vor}_{\mathbb{E}}^d(\mathcal{P}))$ since every Voronoi vertex of $\mathrm{Vor}_{\mathbb{E}}(\mathcal{P})$ is witnessed by the canvas (the contrapositive). Their combination therefore proves Theorem 7.3.1.

The rest of this section is dedicated to proving Lemmas 7.3.2 and 7.3.3; we start by proving the former.

Proof of Lemma 7.3.2

Assume that the Voronoi cell $\mathrm{V}_{\mathbb{E}}(p_0)$ has at least 3 vertices, otherwise we are done as any Voronoi edge of $\mathrm{V}_{\mathbb{E}}(p_0)$ is adjacent to the other Voronoi edges. Let $\{F_i\}$ be the set of faces (edges) of the Voronoi cell $\mathrm{V}_{\mathbb{E}}(p_0)$. We seek a lower bound on the distance between non-adjacent faces

$$d(F_j, F_k) = \min_{x \in F_j, y \in F_k} d_{\mathbb{E}}(x, y),$$

where F_j and F_k are not adjacent. Since a Voronoi cell is a convex polytope, the minimum of this distance will be attained at a vertex of a face. We thus obtain the following result:

Lemma 7.3.4 *Assume that $\mathcal{P}$ is a δ-power protected (ε, μ)-net. Let $\ell = \min_{i,j} d_{\mathbb{E}}(v_i, F_j)$ where F_j is a Voronoi edge whose endpoints are not v_i. The length ℓ can be bounded from below by*

$$\ell \geq \ell_m = \frac{\delta^2 \mu}{8\varepsilon^2} \sqrt{1 - \left(\frac{\mu}{2\varepsilon}\right)^2}.$$

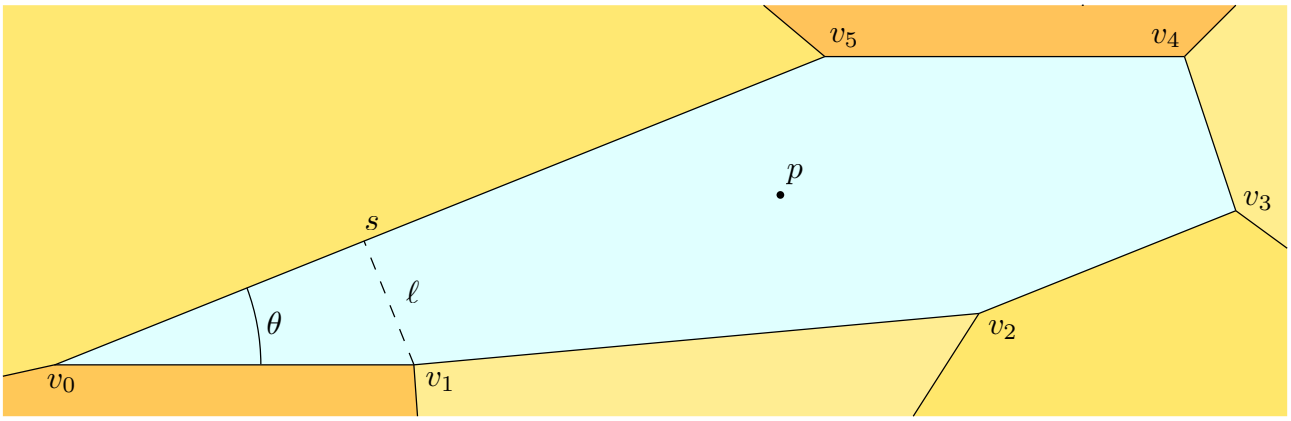

Figure 7.4: Corner of a Voronoi cell. ℓ is the minimal distance between two non-adjacent edges.

Proof. Figure 7.4 shows a generic Euclidean Voronoi cell with 6 Voronoi vertices. Assume without loss of generality that v_1 and $F = [v_0; v_5]$ are the arguments that realize the minimum of the distance and let θ be the angle at v_1. Since we have δ-power protection, the Voronoi vertices of $\mathrm{V}_{\mathbb{E}}(p_0)$ are separated and $|v_0 - v_1| \geq \delta^2/4\varepsilon$ using Lemma 3.2.4 in Chapter 3. If $\theta \geq \pi/2$, we immediately have $\ell \geq \delta^2/4\varepsilon$. Assume now that $\theta \leq \pi/2$. Let $v = |v_0 - v_1|$. By Lemma 3.3.1 in Chapter 3, we have

$$\theta \geq 2 \arcsin\left(\frac{\mu}{2\varepsilon}\right).$$

Therefore,

$$\ell = \sin(\theta) v \geq \sin\left(2 \arcsin\left(\frac{\mu}{2\varepsilon}\right)\right) \frac{\delta^2}{4\varepsilon} = \frac{\delta^2 \mu}{4\varepsilon^2} \sqrt{1 - \left(\frac{\mu}{2\varepsilon}\right)^2},$$

using $\sin(2\alpha) = 2\sin(\alpha)\sqrt{1 - \sin^2(\alpha)}$. Since

$$\frac{\mu}{\varepsilon} \sqrt{1 - \left(\frac{\mu}{2\varepsilon}\right)^2} \leq 1,$$

the minimum is attained when $\theta \leq \pi/2$ and we have

$$\ell_m = \frac{\delta^2 \mu}{4\varepsilon^2} \sqrt{1 - \left(\frac{\mu}{2\varepsilon}\right)^2}.$$

□

As the canvas $\mathcal{C}$ is an isotropic Delaunay mesh driven by a sizing field $h_{\mathcal{C}}$ that bounds the length of the circumradius of a Delaunay simplex, if $\mathcal{C}$ honors a uniform sizing field $h_{\mathcal{C}}$, all the canvas edges e in $\mathcal{C}$ have length $|e| < 2h_{\mathcal{C}}$. Thus if $h_{\mathcal{C}} < \ell_m/2$, then no edge crosses two non-adjacent Voronoi edges. This ensures that for all $D \in \mathcal{N}(\mathrm{Vor}^d_{\mathbb{E}}(\mathcal{P}))$, there exists $D' \in \mathcal{N}(\mathrm{Vor}_{\mathbb{E}}(\mathcal{P}))$ with the same combinatorial information, which proves Lemma 7.3.2.

Proof of Lemma 7.3.3

We now prove Lemma 7.3.3, that is, if $h_{\mathcal{C}}$ is small enough then every Voronoi vertex is witnessed by a at least one canvas simplex.

Since $\mathcal{C}$ is a partition of the domain $\Omega \subset \mathbb{R}^2$, every Voronoi vertex of $\mathrm{Vor}_{\mathbb{E}}(\mathcal{P})$ is located in a maximal canvas simplex $\sigma_{\mathcal{C},i} \in \mathcal{C}$.

Remark 7.3.5 *A Voronoi vertex v_i might be located on a canvas edge (or even a canvas vertex) and belong to more than one canvas simplex. This does not create any issue in the following proofs, and when we refer to the canvas simplex that contains v_i, any canvas simplex that contains the Voronoi vertex can be arbitrarily – but consistently – chosen.*

If the canvas is not dense enough, multiple Voronoi vertices might share a canvas simplex, which is troublesome. Additionally, even if there are no two Voronoi vertices in the same canvas simplex, the canvas simplex $\sigma_{\mathcal{C},i}$ in which v_i is located might not witness v_i. This happens when a canvas edge crosses over two adjacent Voronoi edges, which is not prevented by Lemma 7.3.2. This geometric configuration is illustrated in Figure 7.5.

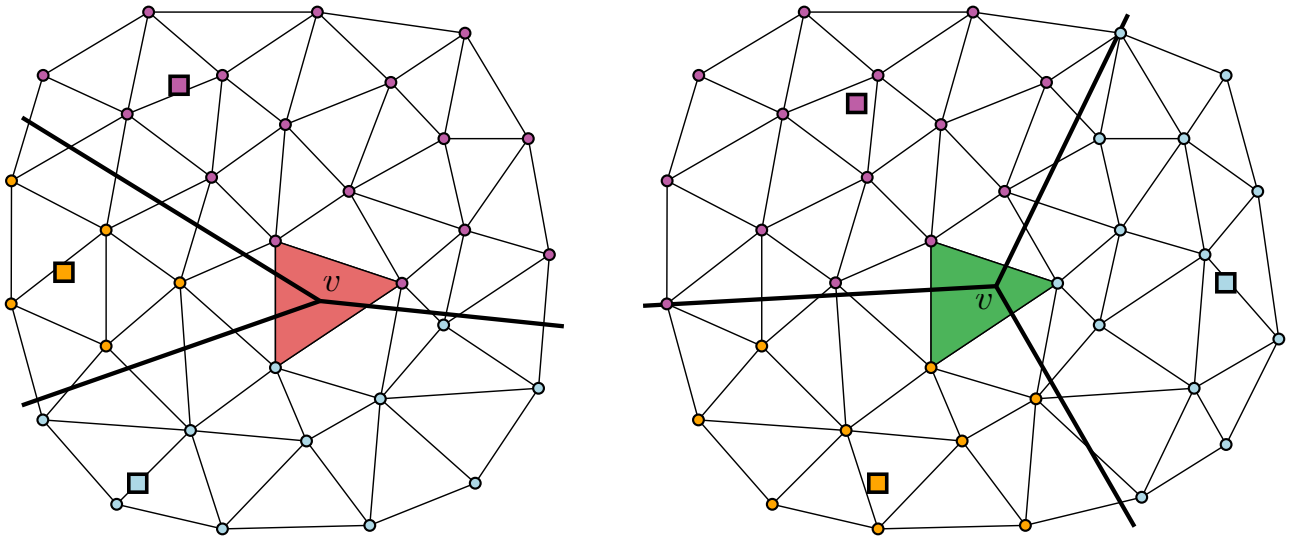

Figure 7.5: Two discrete Riemannian Voronoi diagrams. The canvas is traced with thin black lines, the Riemannian Voronoi diagram with thick black lines. The seeds of the diagram are drawn with squares, and the canvas vertices with circles. On the left, the canvas simplex containing the Voronoi vertex v does not witness the dual of that Voronoi vertex.

The following lemmas prove that these two issues do not appear when the canvas is dense enough.

Lemma 7.3.6 *If $h_{\mathcal{C}} < \delta^2/8\varepsilon$, then no two Voronoi vertices are located in the same maximal canvas simplex.*

Proof. If $h_{\mathcal{C}}$ is the sizing field of $\mathcal{C}$, then any edge of $\mathcal{C}$ has its length bounded by $2h_{\mathcal{C}}$. Since we have assumed δ-power protection of the Delaunay simplices, the Voronoi vertices are separated by $\delta^2/4\varepsilon$ by Lemma 3.2.4 in Chapter 3. Therefore, if $h_{\mathcal{C}} < \delta^2/8\varepsilon$ then every Voronoi vertex falls in a different canvas simplex. □

Recall that for a given v_i, $\sigma_{\mathcal{C},i}$ is the associated canvas simplex that contains v_i. Assume that not every $\sigma_{\mathcal{C},i}$ is also a witness of v_i (otherwise we are done since we have a bijection between $\{v_i\}$ and $\{\sigma_{\mathcal{C},i}\}$ by Lemma 7.3.6). Without loss of generality, we can assume that v_0 is such vertex (hence we assume that $\sigma_{\mathcal{C},0}$ is not a witness of v_0). We will show that if $h_{\mathcal{C}} \leq \ell_m/2$, there exists a canvas simplex that witnesses of v_0, and that we can find it by walking on the canvas as illustrated on Figure 7.6.

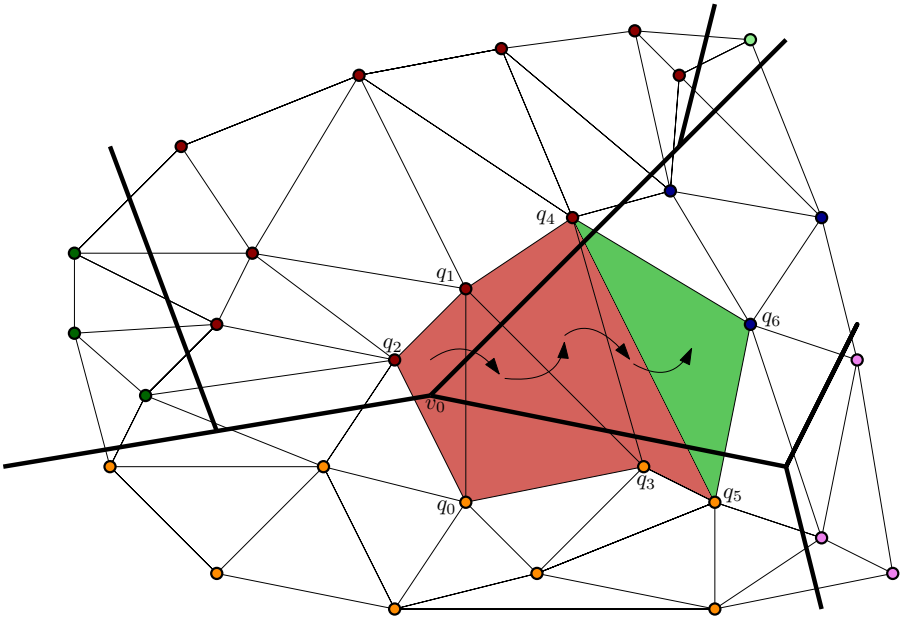

Figure 7.6: The Riemannian Voronoi diagram is traced with thick black lines, the canvas is shown with thin black lines. The algorithm starts in the triangle $\Delta q_0 q_1 q_2$ and eventually finds the witness $\Delta q_4 q_5 q_6$ of v_0.

We now detail the walking algorithm that starts at the simplex $\sigma_{\mathcal{C},0}$ and eventually finds the witness of v_0 in $\mathcal{C}$. An preliminary result is needed to prove that our starting simplex only has two colors.

Lemma 7.3.7 *The vertices of $\sigma_{\mathcal{C},0}$ are colored with exactly two different colors.*

Proof. In this two-dimensional setting, a canvas simplex is colored with at most three different colors. Since we have assumed that $\sigma_{\mathcal{C},0}$ does not witness v_0 and that $h_{\mathcal{C}}$ is small enough such that the Voronoi vertices fall in different canvas simplices, $\sigma_{\mathcal{C},0}$ is not colored with three different colors. The vertices cannot either have all the same colors because v_0 is located in $\sigma_{\mathcal{C},0}$ and Voronoi cells are convex. Therefore, the vertices of $\sigma_{\mathcal{C},0}$ are painted with two different colors. □

Walking process Assume that the conditions of Lemma 7.3.2 are fulfilled. Consequently, no canvas edge can intersect two non-adjacent Voronoi faces. Let q_0, q_1 and q_2 be the vertices of $\sigma_{\mathcal{C},0}$. Without loss of generality, we can assume that q_0 is the vertex that has a different color (see Figure 7.6 for an illustration). Let c_0 and c_1 be the colors at respectively q_0 and q_1. Let c_2 be the color that is "missing"

to capture v_0. Let p_0, p_1 and p_2 be the seeds of the cells V_0, V_1 and V_2 (whose respective colors are c_0, c_1 and c_2).

Begin by picking the edge e_0 of $\sigma_{\mathcal{C},0}$ that is closest to the seed p_2. This is an edge $e_0 = [p_0, p_i]$ with $i = 1$ or 2. Again without loss of generality, assume that $i = 1$. Consider the neighboring canvas simplex $\sigma_{\mathcal{C},1}$ that shares the edge e_0 with $\sigma_{\mathcal{C},0}$. Let q_3 and c_3 be the vertex of $\sigma_{\mathcal{C},1}$ that does not belong to e_0 and its color. There are three possibilities:

- $c_3 = c_2$. We are done: $\sigma_{\mathcal{C},1}$ witnesses v_0.
- $c_3 = c_0$ or $c_3 = c_1$. The same process is repeated: pick the edge of $\sigma_{\mathcal{C},1}$ that is closest to p_2 and has two different colors; find the adjacent σ_2 and look at the color of its third vertex. This will eventually finishes since we move closer to p_2 with every step (and the mesh is finite).
- $c_3 = c_4$ with $c_4 \neq c_0, \ldots, c_2$. This is impossible because it implies that one of the edges of $\sigma_{\mathcal{C},1}$ crosses two non-adjacent Voronoi edges, which contradicts our density assumption.

Therefore, we eventually reach a $\sigma_{\mathcal{C},i}$ that has three different colors, and witnesses v_0. The algorithm is illustrated in Figure 7.6.

This proves Lemma 7.3.3 and ensures that $\forall D \in \mathcal{N}(\mathrm{Vor}_{\mathbb{E}}(\mathcal{P}))$, there exists $D' \in \mathcal{N}(\mathrm{Vor}^d_{\mathbb{E}}(\mathcal{P}))$ with the same combinatorial information. Thus if

$$h_{\mathcal{C}} < \min\left(\frac{\ell_m}{2}, \frac{\delta^2}{8\varepsilon}\right) = \frac{\ell_m}{2} = \frac{\delta^2\mu}{8\varepsilon^2}\sqrt{1 - \left(\frac{\mu}{2\varepsilon}\right)^2},$$

then both lemmas are proven and thus Theorem 7.3.1 is proven, and $\mathcal{N}(\mathrm{Vor}_{\mathbb{E}}(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}^d_{\mathbb{E}}(\mathcal{P}))$.

7.3.2 Uniform metric field

We now consider the setting of $\Omega = \mathbb{R}^2$ endowed with a uniform metric field g_0 and use the results obtained in the Euclidean case to give density conditions on the sampling of the point set and on the canvas such that the nerves of $\mathrm{Vor}^d_{g_0}(\mathcal{P})$ and $\mathrm{Vor}_{g_0}(\mathcal{P})$ are combinatorially equivalent.

By Lemma 3.4.1 in Chapter 3, we have stability of the net and power protection assumption through a metric transformation. That is, if a point set $\mathcal{P}$ in $\mathbb{R}^2$ is a δ-power protected (ε, μ)-net with respect to a uniform metric field g_0 (for all $x \in \Omega$, $g_0(x) = G_0$), then the point set $\mathcal{P}' = \{F_0 p_i, p_i \in \mathcal{P}\}$, where F_0 is the square root of G_0, is a δ-power protected (ε, μ)-net with respect to the Euclidean metric. We can deduce from this result an upper bound on the sizing field of $\mathcal{C}$ in the case of a uniform metric field using the results of Section 7.3.1 in the Euclidean setting.

Theorem 7.3.8 *Let g_0 be a uniform metric field on Ω (for all $x \in \Omega$, $g_0(x) = G_0$). Assume that $\mathcal{P}$ is a δ-power protected $(\varepsilon, \mu)-$net in Ω with respect to g_0. Denote by $\mathcal{C}$ the canvas, and $h_{\mathcal{C}}$ its sizing field. Let $\{\lambda_i\}$ and $\{v_i\}$ be respectively the eigenvalues and the normalized eigenvectors of the matrix G_0. If*

$$h_{\mathcal{C}} < \left(\min_i \sqrt{\lambda_i}\right)\left(\frac{\delta^2\mu}{8\varepsilon^2}\sqrt{1 - \left(\frac{\mu}{2\varepsilon}\right)^2}\right),$$

then $\mathcal{N}(\mathrm{Vor}_{g_0}(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}^d_{g_0}(\mathcal{P}))$.

Proof. Consider $\mathcal{R} = \mathrm{Vor}^d_{g_0}(\mathcal{P})$ over $\mathcal{C}$. Let F_0 be a square root of G_0. The matrix F_0 provides a stretching operator between the Euclidean and the metric spaces. Let $\mathcal{P}_0 = \{F_0 p, p \in \mathcal{P}\}$ be the

transformed point set and $\mathcal{C}_0$ be a canvas (an isotropic Delaunay triangulation) of the transformed space. Denote by $\mathcal{R}_0$ the discrete Riemannian Voronoi diagram of $\mathcal{P}_0$ with respect to $g_{\mathbb{E}}$ over $\mathcal{C}_0$. Let $h_{\mathcal{C}_0}$ be the upper bound on the sizing field of $\mathcal{C}_0$ provided by Theorem 7.3.1 such that $\mathrm{Del}_{\mathbb{E}}(\mathcal{P}_0)$ is captured by $\mathcal{C}_0$.

Since $\mathcal{P}_0$ is a δ-power protected net with respect to $g_{\mathbb{E}}$, we can invoke Theorem 7.3.1 and we must thus have

$$h_{\mathcal{C}_0} < \frac{\delta^2\mu}{8\varepsilon^2}\sqrt{1-\left(\frac{\mu}{2\varepsilon}\right)^2},$$

for the canvas $\mathcal{C}_0$ to capture $\mathcal{N}(\mathrm{Vor}_{\mathbb{E}}(\mathcal{P}_0))$.

Let $\mathcal{C}_0'$ be the image of $\mathcal{C}$ by F_0. Note that $\mathcal{C}_0'$ and $\mathcal{C}_0$ are two different triangulations of the same space. If any edge of $\mathcal{C}_0'$ is smaller (with respect to the Euclidean metric) than $2h_{\mathcal{C}_0}$, then $\mathcal{C}_0'$ satisfies

$$h_{\mathcal{C}_0'} < \frac{\delta^2\mu}{8\varepsilon^2}\sqrt{1-\left(\frac{\mu}{2\varepsilon}\right)^2}$$

and thus $\mathcal{C}_0'$ captures $\mathcal{N}(\mathrm{Vor}_{\mathbb{E}}(\mathcal{P}_0))$.

Recall that given an eigenvector v_i of G_0 with corresponding eigenvalue λ_i, a unit length in the direction v_i in the metric space has length $1/\sqrt{\lambda_i}$ in the Euclidean space. Therefore, if the sizing field $h_{\mathcal{C}}$ of $\mathcal{C}$ is smaller than $\alpha h_{\mathcal{C}_0}$, with

$$\alpha = \frac{1}{\max\limits_i\left(\frac{1}{\sqrt{\lambda_i}}\right)} = \min_i \sqrt{\lambda_i},$$

then every edge of $\mathcal{C}_0'$ is smaller than $2h_{\mathcal{C}_0}$. This implies that $\mathcal{C}_0'$ captures $\mathcal{N}(\mathrm{Vor}_{\mathbb{E}}(\mathcal{P}_0))$ and therefore $\mathcal{C}$ captures $\mathcal{N}(\mathrm{Vor}_{g_0}(\mathcal{P}))$. □

This settles the case of a uniform metric field.

7.3.3 Arbitrary metric field

We now consider an arbitrary metric field g over the domain $\Omega = \mathbb{R}^2$. The key to proving the equivalence in this setting is to locally approximate the arbitrary metric field with a uniform metric field, a configuration that we have dealt with in the previous section. We shall always compare the metric field g in a neighborhood U with the uniform metric field $g' = g(p_0)$ where $p_0 \in U$. Because g' and the Euclidean metric field differ by a linear transformation, we can simplify matters and assume that g' is the Euclidean metric field $g_{\mathbb{E}}$. The main argument of the proof will be once again that a power protected net has stable and separated Voronoi vertices.

We now introduce the main result of this section, Theorem 7.3.9.

Theorem 7.3.9 *Let g be an arbitrary metric field on Ω. Assume that $\mathcal{P}$ is a δ-power protected $(\varepsilon,\mu)-$net in Ω with respect to g. Denote by $\mathcal{C}$ the canvas, and $h_{\mathcal{C}}$ its sizing field. If*

$$h_{\mathcal{C}} < \min_{p\in\mathcal{P}} h_{\mathcal{C},p},$$

where $h_{\mathcal{C},p}$ is given in Lemma 7.3.10 below, and if ε is sufficiently small and δ is sufficiently large (both values will be detailed in the proof), then $\mathcal{N}(\mathrm{Vor}_g(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P}))$.

We will prove Theorem 7.3.9 by computing for each point $p \in \mathcal{P}$ the necessary sizing field of the canvas such that the Voronoi cell $\mathrm{V}_g(p)$ is captured correctly. Conditions on ε and δ shall emerge from the intermediary results on the stability of power protected nets.

Lemma 7.3.10 *Let $\psi_0 \geq 1$ be a bound on the metric distortion and g be a Riemannian metric field on U. Let U be an open neighborhood of $\Omega = \mathbb{R}^2$ that is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$ such that $\forall p \in B(p_0, r_0), \psi(g(p), g_{\mathbb{E}}(p)) \leq \psi_0$. Let $\mathcal{P}_U$ be a finite point set in U and let $p_0 \in \mathcal{P}_U$.*

Suppose that $\mathcal{P}_U$ is a δ-power protected $(\varepsilon, \mu)-$net of with respect to g. Let $\mathrm{V}_g(p_0)$ be the Voronoi cell of p_0 in $\mathrm{Vor}_g^d(\mathcal{P}_U)$.

If

$$h_{\mathcal{C},p_0} < \frac{1}{2} \min_i \left(\sqrt{\lambda_i}\right) \sin(\theta) \left(L_0 - \frac{2\eta}{\tan\left(\frac{\theta}{2}\right)}\right),$$

with $\{\lambda_i\}$ the eigenvalues of g_0, and if ε is sufficiently small and δ is sufficiently large (both values will be detailed in the proof), then the cell $\mathrm{V}_g(p_0)$ is captured.

Approach

Our approach is composed of essentially two steps: we first bring ourselves back to a setting that we have solved and then use the previous results. For readability, the many intermediary results needed to prove Theorem 7.3.9 are moved to Chapter 3. We refer to them at appropriate times.

We consider a seed $p \in \mathcal{P}$ and first prove that the Voronoi cell $\mathrm{V}_g(p)$ is close to the Euclidean Voronoi cell $\mathrm{V}_{\mathbb{E}}(p)$. Specifically, we show (Lemma 3.4.12 in Chapter 3) that the Riemannian Voronoi cell can be encompassed into two scaled versions of the Euclidean Voronoi cell: an eroded cell $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p)$ and a dilated cell $\mathrm{DV}_{\mathbb{E}}^{+\eta}(p)$, defined respectively by

$$\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0) = \{x \in \mathrm{V}_{\mathbb{E}}(p_0) \mid d_{\mathbb{E}}(x, \partial\, \mathrm{V}_{\mathbb{E}}(p_0)) > \eta\},$$

and

$$\mathrm{DV}_{\mathbb{E}}^{+\eta}(p_0) = \bigcap_{i \neq 0} H^{\eta}(p_0, p_i),$$

where $H^{\eta}(p_0, p_i)$ is the half-space containing p_0 and delimited by the bisector $\mathrm{BS}(p_0, p_i)$ translated away from p_0 by η. The constant η is the thickness of this encompassing and depends on the bound on the distortion ψ_0 in the neighborhood, and on the sampling and separation parameters ε and μ. We have that η goes to 0 as ψ_0 goes to 1. Figure 7.7 illustrates the notations.

We then show that if $\mathcal{P}$ is a power protected net with respect to g then $\mathcal{P}$ is locally a power protected net with respect to $g_{\mathbb{E}}$ (Lemmas 3.4.2 and 3.4.16 in Chapter 3). From this conclusion, the separation and stability of the Voronoi vertices of $\mathrm{V}_{\mathbb{E}}(p)$ can be obtained. We then expose conditions on the quality of the seed set such that the Voronoi vertices of $\mathrm{EV}_g^{-\eta}(p)$ are separated (Lemma 7.3.12 of this chapter).

Remark 7.3.11 *There is so far no involvement of the canvas whatsoever, and hence this procedure can be almost exactly used to prove the embedding of the dual of the anisotropic Voronoi diagram of Labelle and Shewchuk (see Chapter II.5).*

Finally, we expose conditions such that $\mathrm{EV}_g^{-\eta}(p)$ – and thus $\mathrm{V}_g(p)$ – is captured by the canvas through density requirements. Considering the minimum for all points $p \in \mathcal{P}$, we obtain the final metric field.

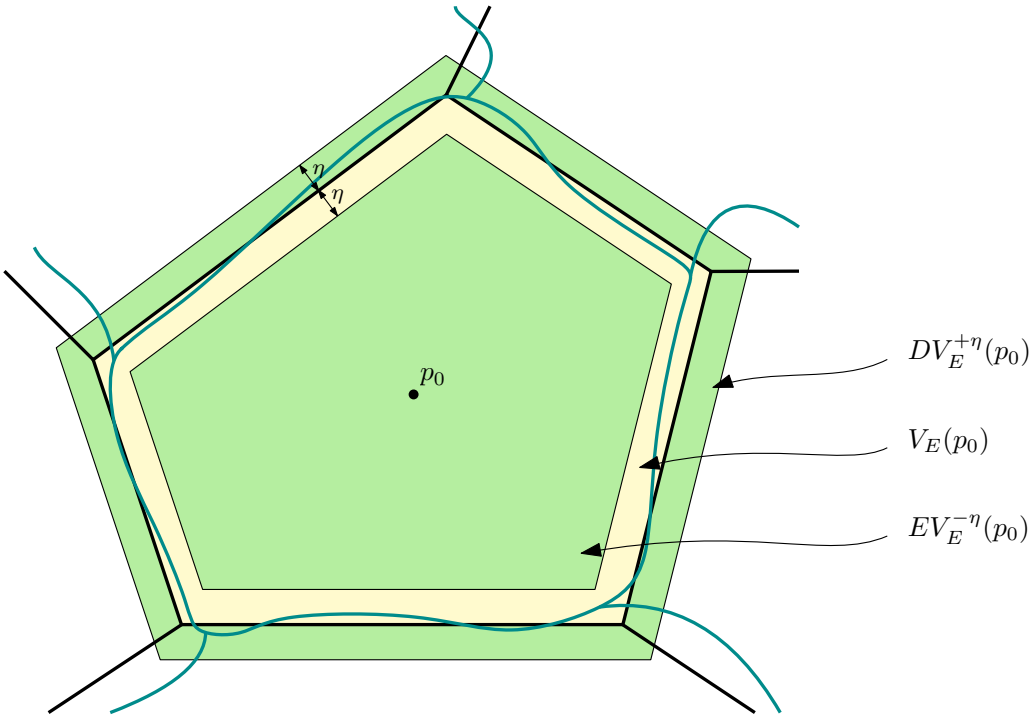

Figure 7.7: The different encompassing cells around p_0. The Riemannian Voronoi diagrams with respect to g and $g_{\mathbb{E}}$ are respectively traced in dark cyan and black. The Voronoi cell $\mathrm{V}_{\mathbb{E}}(p_0)$ is colored in yellow. The cells $\mathrm{DV}_{\mathbb{E}}^{+\eta}$ and $\mathrm{EV}_{\mathbb{E}}^{-\eta}$ are colored in green.

Proof of Lemma 7.3.10

As recalled above, Lemma 3.4.5 in Chapter 3 gives us that $\mathrm{V}_g(p_0)$ lies in $\mathrm{DV}_{\mathbb{E}}^{+\eta}(p_0)$ and contains $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$. Since $\mathrm{V}_g(p_0)$ contains $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$, if $h_{\mathcal{C}}$ is small enough such that $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$ is captured by $\mathcal{C}$, then $\mathrm{V}_g(p_0)$ is also captured.

In the Euclidean setting (see Section 7.3.1), we have used the power protection of $\mathcal{P}$ with respect to $g_{\mathbb{E}}$ to deduce a lower bound on the distance between two adjacent Voronoi vertices (see Lemma 3.2.4 in Chapter 3). Since we know that $\mathcal{P}$ is a power protected net for $g_{\mathbb{E}}$ by stability, we can apply the separation bound to the vertices of $\mathrm{V}_{\mathbb{E}}(p_0)$, but not to those of $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$. Nevertheless, we can still extract the critical property that the Voronoi vertices are separated, as shown in the next lemma.

Lemma 7.3.12 (Separation of the Voronoi vertices of the eroded Voronoi cell) *Let U, g and ψ_0 be defined as in Lemma 7.3.10. Assume that the point set $\mathcal{P}_U$ is a δ-power protected (ε, μ)-net with respect to the Riemannian metric field g. Then the Voronoi vertices of $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$ are separated, under sufficient conditions on δ and ε. (Those values will be made explicit in the proof.)*

Proof. The notations used in this proof are illustrated in Figure 7.8. By Lemmas 3.4.2 and 3.4.16, $\mathcal{P}$ is δ_0-power protected (ε_0, μ_0)-net with respect to $g_{\mathbb{E}}$. The δ_0-power protection property of $\mathcal{P}_U$ induces a separation bound $L_0 := \delta_0^2/4\varepsilon_0$ between the Voronoi vertices of $\mathrm{V}_{\mathbb{E}}(p_0)$ (see Lemma 3.2.4). Denote by ℓ be the distance between any two adjacent Voronoi vertices of $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$. When a seed set is a net, we can deduce a bound on the angle of a corner of the Voronoi cell $\mathrm{V}_{\mathbb{E}}(p_0)$ (using Lemma 3.3.1 in Chapter 3); denote this bound by $\theta = 2\arcsin(\mu_0/2\varepsilon_0)$. Note that the lower bound on the smallest angle θ in a Voronoi cell is unchanged by the erosion of the Voronoi cell.

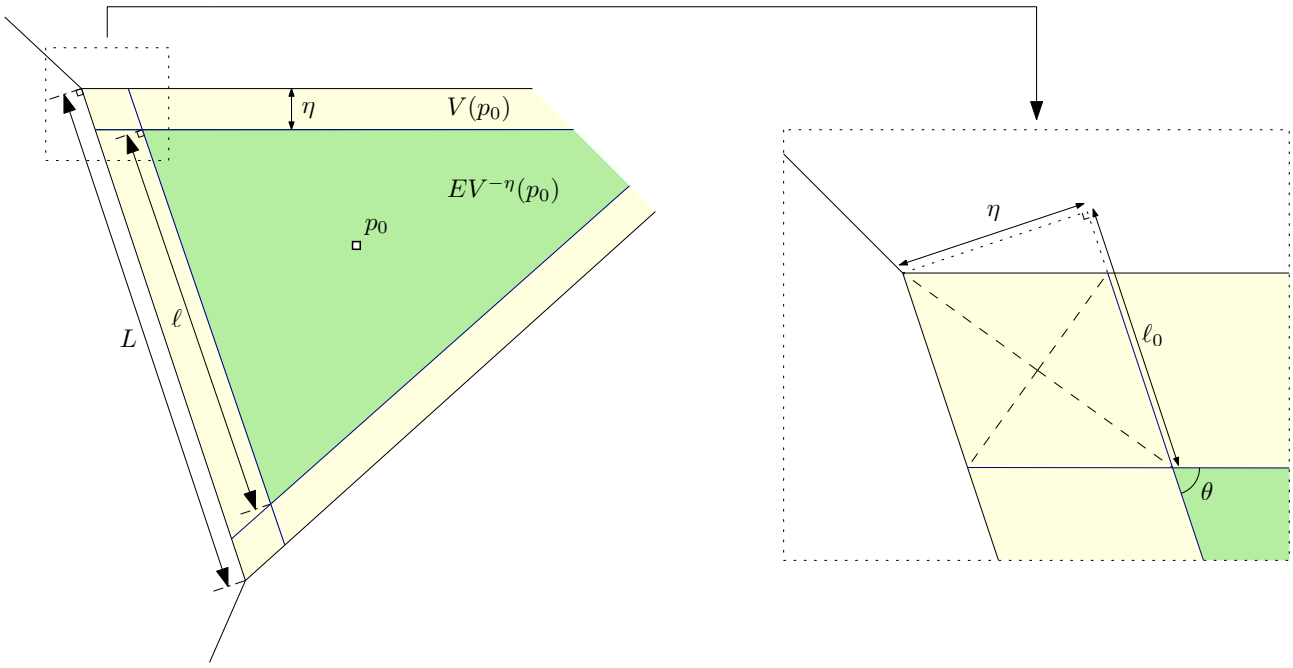

Figure 7.8: Eroded Voronoi cell and a zoom on one of the corners.

We have

$$\ell = L - 2\ell_0, \text{ with } \ell_0 = \frac{\eta}{\tan\left(\frac{\theta}{2}\right)}.$$

Observing that

$$\tan\left(\arcsin(x)\right) = \frac{x}{\sqrt{1-x^2}} = \frac{1}{\sqrt{\frac{1}{x^2}-1}},$$

we obtain

$$\begin{aligned}
\ell &= L - 2\frac{\eta}{\tan\left(\frac{\theta}{2}\right)} \\
&\geq L_0 - 2\frac{\eta}{\tan\left(\arcsin\left(\frac{\mu_0}{2\varepsilon_0}\right)\right)} \\
&= L_0 - 2\eta\sqrt{\left(\frac{2\varepsilon_0}{\mu_0}\right)^2 - 1} \\
&\geq \frac{\delta_0^2}{4\varepsilon_0} - 2\frac{(2\varepsilon_0)^2(\psi_0^2-1)}{\mu_0}\sqrt{\left(\frac{2\varepsilon_0}{\mu_0}\right)^2 - 1},
\end{aligned} \tag{7.1}$$

since $\eta = \frac{\rho_0^2((2\varepsilon_0)^2-1)}{\mu_0}$ (see Definition 3.4.9 and Lemma 3.4.15 in Chapter 3). To have separation of the Voronoi vertices of $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$, we need ℓ to be positive. This can be achieved by enforcing that the lower bound is positive:

$$\frac{\delta_0^2}{4\varepsilon_0} > 2\frac{(2\varepsilon_0)^2(\psi_0^2-1)}{\mu_0}\sqrt{\left(\frac{2\varepsilon_0}{\mu_0}\right)^2 - 1}.$$

By Lemma 3.4.2, we have that $\varepsilon_0 = \psi_0\varepsilon$, $\mu_0 = \lambda\varepsilon/\psi_0$ and $\rho_0 = 2\psi_0\varepsilon$. Using this notation we see that $\ell > 0$ if

$$\delta_0^2 > \frac{32}{\lambda}(\psi_0^2-1)\varepsilon^2\psi_0^4\sqrt{\left(\frac{2\psi_0^2}{\lambda}\right)^2 - 1}. \tag{7.2}$$

This condition is easy to satisfy when ψ_0 goes to 1, because the right hand side of Inequality (7.2) is proportional to $(\psi_0^2-1)\varepsilon^2$.

The intermediary results that we use already impose some conditions on δ and ε, and we thus would like to give the condition in Equation 7.2 in terms of δ, so that it may be compared with Inequality (3.8) (this will be done in Remark 7.3.13). In Lemma 3.4.16, we have seen that

$$\delta_0^2 = \frac{\delta^2}{\psi_0^2} + \left(\frac{1}{\psi_0^2} - \psi_0^2\right)(\varepsilon+\chi_2)^2 - \frac{4\varepsilon\chi_2}{\psi_0^2},$$

with, by Lemma 3.4.13 in Chapter 3,

$$\chi_2 = \frac{8\psi_0^3\varepsilon(\psi_0^2-1)}{\lambda\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)}.$$

Thus

$$\frac{\delta^2}{\psi_0^2} + \left(\frac{1}{\psi_0^2} - \psi_0^2\right)(\varepsilon+\chi_2)^2 - \frac{4\varepsilon\chi_2}{\psi_0^2} > \frac{32}{\lambda}(\psi_0^2-1)\varepsilon^2\psi_0^4\sqrt{\left(\frac{2\psi_0^2}{\lambda}\right)^2 - 1},$$

which is equivalent to

$$\delta^2 > \frac{32}{\lambda}(\psi_0^2-1)\varepsilon^2\psi_0^6\sqrt{\left(\frac{2\psi_0^2}{\lambda}\right)^2 - 1} - \left(1-\psi_0^4\right)(\varepsilon+\chi_2)^2 + 4\varepsilon\chi_2.$$

After simplifying the $(\varepsilon+\chi_2)$-term, we find

$$\delta^2 > \frac{32}{\lambda}(\psi_0^2-1)\varepsilon^2\psi_0^6\sqrt{\left(\frac{2\psi_0^2}{\lambda}\right)^2 - 1} + \frac{32\psi_0^3(\psi_0^2-1)\varepsilon^2}{\lambda\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)} + \left(\psi_0^2-1\right)\varepsilon^2\left(\psi_0^2+1\right). \tag{7.3}$$

When this bound is satisfied, the Voronoi vertices of $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$ are separated, which concludes our proof. □

Remark 7.3.13 *Proving Lemma 3.4.12 in Chapter 3 required us to prove various intermediary lemmas, which already introduced requirements on the sampling. If we compare Equation 7.3 to the bound given by Inequality* (3.8) *(made explicit in Inequality* (3.9)*) in the proof of Lemma 3.4.16, namely*

$$\begin{aligned}\delta^2 >& \frac{256(\psi_0^2-1)\psi_0^2}{\lambda^2\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)^2}(\psi_0^2-1)\varepsilon^2 \\ &+ \frac{32\psi_0^3}{\lambda\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)}(\psi_0^2-1)\varepsilon^2 \\ &+ (\psi_0^2+1)(\psi_0^2-1)\varepsilon^2.\end{aligned} \tag{7.4}$$

We see that the two final terms in the sum in Inequalities (7.3) *and* (7.4) *are of equal magnitude. The final terms would in fact agree if we would not have simplified the equation, because* χ_2 *is of order* $\varepsilon(\psi_0^2 - 1)$*. The first term disappears for Inequality* (7.4) *in the limit where* ψ_0 *tends to* 1*, while this is not the case for the first term in Inequality* (7.3)*.*

This means that – at least when ψ_0 *goes to* 1 *– Equation 7.3 provides the tougher bound.*

We can now provide a bound on the sizing field $h_{\mathcal{C}}$ of the canvas such that $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$ is captured and prove Lemma 7.3.10. Similarly to the Euclidean case (see Theorem 7.3.1 and Lemmas 7.3.2 and 7.3.3), the bound on the sizing field is computed by combining the separation distance ℓ (given in Equation 7.1) and the lower bound θ on the angle of a corner of an Euclidean Voronoi cell, giving the requirement

$$h_{\mathcal{C},p_0} < \frac{1}{2}\sin(\theta)\left(L_0 - \frac{2\eta}{\tan\left(\frac{\theta}{2}\right)}\right).$$

We should not forget that we have assumed that g_0 is the Euclidean metric field, which is generally not the case, we must in fact proceed like for the case of a uniform metric field (Theorem 7.3.8):

$$h_{\mathcal{C},p_0} < \frac{1}{2}\min_i\left(\sqrt{\lambda_i}\right)\sin(\theta)\left(L_0 - \frac{2\eta}{\tan\left(\frac{\theta}{2}\right)}\right),$$

with $\{\lambda_i\}$ the eigenvalues of g_0.

Therefore, if the seed set satisfies the previous conditions on ε and δ and the canvas honors the sizing field $h_{\mathcal{C}}$, then $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$ is captured, and thus $\mathrm{V}_g(p_0)$ is captured, which proves Lemma 7.3.10.

Proof of Theorem 7.3.9

Taking the minimum of all the sizing field values over all $p \in \mathcal{P}$, we obtain a sizing field such that all the Voronoi cells are captured. If the canvas $\mathcal{C}$ satisfies

$$h_{\mathcal{C}} < \min_{p\in\mathcal{P}}\left[\frac{1}{2}\min_{\lambda_i\in\mathrm{ev}(g(p))}\left(\sqrt{\lambda_i}\right)\sin(\theta)\left(L_0 - \frac{2\eta}{\tan\left(\frac{\theta}{2}\right)}\right)\right],$$

where $\mathrm{ev}(g(p))$ are the eigenvalues of the matrix $G_p = g(p)$, then every cell is captured properly and $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}_g(\mathcal{P}))$ in the general setting, when the geodesics are exactly computed.

7.3.4 Extension to surfaces

The previous results may be generalized to surfaces embedded in $\mathbb{R}^3$. We shall assume that, apart from the metric induced by the embedding of the domain in Euclidean space, there is a second metric g defined on $\mathcal{M}$. Let $\pi_p : \mathcal{M} \to T_p\mathcal{M}$ be the orthogonal projection of points of $\mathcal{M}$ on the tangent space $T_p\mathcal{M}$ at p. For a sufficiently small neighborhood $U_p \subset T_p\mathcal{M}$, π_p is a local diffeomorphism (see Niyogi [111]).

Denote by $\mathcal{P}_{T_p}$ the point set $\{\pi_p(p_i), p_i \in \mathcal{P}\}$ and $\mathcal{P}_{U_p}$ the restriction of $\mathcal{P}_{T_p}$ to U_p. Assuming that the conditions of Niyogi [111] are satisfied (which are simple density constraints on ε compared to the reach of the manifold), the pullback of the metric with the inverse projection $(\pi_p^{-1})^* g$ defines a metric g_p on U_p such that for all $q, r \in U_p$,

$$d_{g_p}(q, r) = d_g(\pi_p^{-1}(q), \pi_p^{-1}(r)).$$

This implies immediately that if $\mathcal{P}$ is a δ-power protected (ε, μ)-net on $\mathcal{M}$ with respect to g then $\mathcal{P}_{U_p}$ is a δ-power protected (ε, μ)-net on U_p. We have thus a metric on a subset of a two-dimensional space, in this case the tangent space, giving us a setting that we have already solved.

It is left to translate the sizing field requirement from the tangent plane to the manifold $\mathcal{M}$ itself. Note that the transformation π_p is completely independent of g. Boissonnat et al. [17, Lemma 3.7] give bounds on the metric distortion of the projection on the tangent space. This result allows to carry the canvas sizing field requirement from the tangent space to $\mathcal{M}$.

7.3.5 Approximate geodesic distance computations

We have so far assumed that geodesics are computed exactly, which is generally not the case in practice. Nevertheless, once the error in the approximation of the geodesic distances is small enough, the computation of the discrete Riemannian Voronoi diagram with approximate geodesic distances can be equivalently seen as the computation of a second discrete Riemannian Voronoi diagram using exact geodesic distances but for a slightly different metric field.

Denote by $\widetilde{d_g}$ the geodesic approximation and d_g the exact geodesic distance with respect to the metric field g. Assume that in a small enough neighborhood U (see Lemma 3.1.4 in Chapter 3), the distances can be related as

$$\left| d_g(p_0, x) - \widetilde{d_g}(p_0, x) \right| \leq \xi d_g(p_0, x),$$

where ξ is a function of x that goes to 0 as the sampling parameter ε goes to 0. We can formulate a lemma similar to Lemma 3.4.5 in Chapter 3 to bound the distance between the same bisectors between seeds for the exact and the approximate diagrams.

Lemma 7.3.14 *Let $\psi_0 \geq 1$ be a bound on the metric distortion and g and g' be two Riemannian metric fields on U. Let U be an open neighborhood of $\Omega = \mathbb{R}^2$ that is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$ such that $\forall p \in B(p_0, r_0), \psi(g(p), g'(p)) \leq \psi_0$. Let $\mathcal{P}_U = \{p_i\}$ be a finite point set in U and let $p_0 \in \mathcal{P}_U$. Let $\widetilde{V_{p_0,g}}$ denote a Voronoi cell with respect to the approximate geodesic distance.*

Suppose that the Voronoi cell $V_g^{+2\xi\tilde{\rho}}(p_0)$ lies in a ball of radius $\tilde{\rho}$ with respect to the metric g, which lies completely in U. Then $\widetilde{V_{p_0,g}}$ lies in $V_g^{+2\xi\tilde{\rho}}(p_0)$ and contains $V_g^{-2\xi\tilde{\rho}}(p_0)$.

Proof. Let $\widetilde{\mathrm{BS}_g}(p_0, p_i)$ be the bisector between p_0 and p_i with respect to the approximate geodesic distance. Let $y \in \widetilde{\mathrm{BS}_g}(p_0, p_i) \cap B_g(p_0, \rho)$, where $B_g(p_0, \rho)$ denotes the ball centered at p_0 of radius ρ with respect to the exact geodesic distance. Now $\widetilde{d_g}(y, p_0) = \widetilde{d_g}(y, p_i)$, and thus

$$\begin{aligned}
|d_g(y,p_0)^2 - d_g(y,p_i)^2| &= \left| d_g(y,p_0)^2 - \widetilde{d_g}(y,p_0)^2 + \widetilde{d_g}(y,p_i)^2 - d_g(y,p_i)^2 \right| \\
&\leq \left| d_g(y,p_0)^2 - \widetilde{d_g}(y,p_0)^2 \right| + \left| d_g(y,p_i)^2 - \widetilde{d_g}(y,p_i)^2 \right| \\
&\leq 2\xi (d_g(y,p_0)^2 + d_g(y,p_i))^2 \\
&\leq 2\xi\tilde{\rho}^2.
\end{aligned}$$

Thus $d_g(y,p_0)^2 \leq d_g(y,p_i)^2 + \tilde{\omega}$ and $d_g(y,p_0)^2 \geq d_g(y,p_i)^2 - \tilde{\omega}$ with $\tilde{\omega} = 2\xi\tilde{\rho}^2$, which gives us the expected result. □

Denote by $\widetilde{V}(p_0)$ the Voronoi cell of p_0 with the approximate metric. We can then incorporate Lemma 7.3.14 to obtain a result similar to Lemma 3.4.12.

Lemma 7.3.15 *Let U, ψ_0, g, g_0 and $\mathcal{P}_U$ be defined as in Theorem 7.3.9. Then we can find η_0' and ω_0' such that*

$$\mathrm{EV}_{g_0}^{-\eta_0'}(p_0) \subseteq \mathrm{V}_{g_0}^{-\omega_0'}(p_0) \subseteq \mathrm{V}_{g}^{-\tilde{\omega}}(p_0) \subseteq \widetilde{\mathrm{V}}_{g}(p_0) \subseteq \mathrm{V}_{g}^{+\tilde{\omega}}(p_0) \subseteq \mathrm{V}_{g_0}^{+\omega_0'}(p_0) \subseteq \mathrm{DV}_{g_0}^{+\eta_0'}(p_0).$$

These inclusions are illustrated in Figure 7.9.

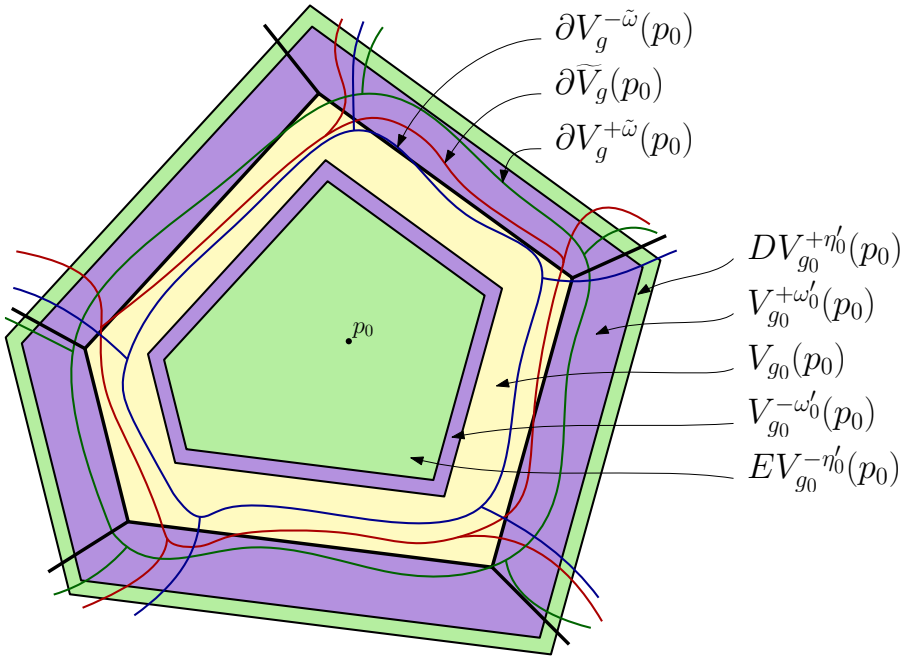

Figure 7.9: Illustration of the different encompassing cells around p_0 in the context of an approximate geodesic distance. The RVDs with respect to g and g_0 are respectively traced in red and black. The dilated Voronoi cells are traced in blue and green. The Voronoi cell $V_{g_0}(p_0)$ is colored in yellow. The cells $DV_{g_0}^{+\eta_0'}$ and $EV_{g_0}^{-\eta_0'}$ are colored in green, and the cells $V_{g_0}^{\pm\omega_0'}(p_0)$ are colored in purple.

The subsequent lemmas and proofs are similar to what was done in the case of exact geodesic computations and we do not detail them.

7.4 Curved Riemannian Delaunay triangulation

In the previous section, we have given sufficient conditions for the nerves of the discrete and Riemannian Voronoi diagrams to be equivalent. The next step is to prove that the curved Riemannian Delaunay complex is homeomorphic to the manifold. Several works have been dedicated to conditions that guarantee this, see for example Edelsbrunner and Shah [69] and Boissonnat et al. [19, 20]. We shall be relying on the latter ones for many results.

In two dimensions, the Delaunay complex is homeomorphic to the surface under very light conditions: it is sufficient for three points not to lie on a geodesic, see Rustamov [124], Dyer et al. [65], or Wintraecken [140]. We prove the same results using a more geometrical approach where we "paint" the Delaunay simplex on the manifold in an intrinsic manner and, along the way, fix a flaw in Dyer et al. [67]. The main result is given in Theorem 7.4.18.

For each simplex σ in the Delaunay complex, we consider its curved realization (see Section 2.9 in Chapter 2 for the exact definition). As in the work of Dyer [65], we can define the non-degeneracy of a geometric simplex as follows

Definition 7.4.1 *The curved realization of a Delaunay simplex, $\widetilde{\sigma}$, is said to be non-degenerate if, and only if, it is homeomorphic to $\boldsymbol{\psi}$, where $\boldsymbol{\psi}^n$ is the n-dimensional standard simplex with coordinates $\lambda = (\lambda_i)$, that is*

$$\boldsymbol{\psi}^n = \{(\lambda_i) \in \mathbb{R}^{n+1} \mid \lambda_i \geq 1, \sum_i \lambda_i = 1\}.$$

We are interested in conditions that guarantee that the curved realization of the Delaunay complex gives a homeomorphism between the Delaunay complex Del and the manifold $\mathcal{M}$. The homeomorphism is in fact a piecewise smooth homeomorphism where the homeomorphism per simplex $\widetilde{\sigma}$ is given by the maps $\mathcal{B}_\sigma$. It is of course necessary that all simplices $\widetilde{\sigma}$ in the complex are non-degenerate.

We first recall some definitions:

Definition 7.4.2 *A mapping f of an oriented simplicial complex K that is a n-manifold (with boundary), into Euclidean space $\mathbb{E}^n$ endowed with an orientation, is called simplex-wise positive if for each top dimensional simplex σ_i^n, f is smooth and bijective in σ_i^n (the partial derivatives being continuous on the boundary), as well as orientation preserving. For any simplicial complex L, let L^k, or $(L)^k$, denote the subcomplex containing all cells of L of dimension $\leq k$. With f in K as above, any point q of $\mathbb{E}^n \setminus f(K^{n-1})$ is in the image of a certain number of h of n-simplices of K; we say that q is covered h times.*

Lemma 7.4.3 (Whitney's lemma [139]) *Let f be simplex-wise positive in K. Then for any connected open subset R of $\mathbb{E}^n \setminus f(\partial K)$, any two points of R not in $f(K^{n-1})$ are covered the same number of times. If this number is 1, then f, considered in the open subset $R' = f^{-1}(R)$ of K only, is bijective onto R.*

We recall some definitions and a lemma introduced by Leibon [90] and Dyer [67].

Definition 7.4.4 (Strong convexity radius) *A geodesic disk $B_r(x)$ centered at x on a Riemannian surface is called strongly convex if for any two point in the closed disk $\bar{B}_r(x)$ there is exists a unique minimizing geodesic connecting these points which lies in the interior of $B_r(x)$. The strong convexity radius is the largest radius for which the disk is strongly convex.*

We have the following bound on the convexity radius by Chavel [41, Theorem IX.6.1]:

Lemma 7.4.5 *Suppose the sectional curvatures of $\mathcal{M}$ are bounded by $K \leq \Lambda_+$, and $i_{\mathcal{M}}$ is the injectivity radius. If*

$$r < \min\left\{\frac{i_{\mathcal{M}}}{2}, \frac{\pi}{2\sqrt{\Lambda_+}}\right\},$$

then $\bar{B}_{\mathcal{M}}(x; r)$ is convex. (If $\Lambda_+ \leq 0$, we take $1/\sqrt{\Lambda_+}$ to be infinite.)

Remark 7.4.6 *If the vertices of an intrinsic simplex lie inside a convex disk, then the simplex is wholly contained in the convex disk.*

Definition 7.4.7 (Pseudo-disks – Boissonnat and Oudot [27]) *A family of topological disks on a surface are Pseudo-Disks if for any two distinct disks, their boundary either do not intersect, or they intersect tangentially at a single point, or they intersect transversely at exactly two points.*

Lemma 7.4.8 (Small circle intersection – Dyer et al. [67], Leibon [90]) *All geodesic disks whose radius is strictly smaller than the strong convexity radius are pseudo-disks.*

We assume that there are at least two intersection points of the two circles, and want to prove that there are not more than 2. We refer to the work of Dyer et al. [67] for the other cases. However, this is precisely where the proof of [67] has a hole. Indeed, it is claimed that given two convex bodies C_1 and C_2 in dimension 2, the boundary of $\partial C_1 \cap \partial C_2$ consists of 2 connected arcs, one on ∂C_1 and one on ∂C_2. This is not true as can be seen by taking two ellipses and rotating one by π.

Proof. Assume that two geodesic circles of radius r and 1 intersect twice on one side of the extended geodesic between the two centers c_1 and c_2. Denote the intersection points by A and B. Two of the geodesics between A and B and c_1 and c_2 respectively must intersect in the point p, because of the uniqueness of geodesics. Without loss of generality, we can assume that Ac_2 and Bc_1 intersect. Denote the lengths of the geodesics Ap by a_1, the length pc_2 by a_2, the length Bp by b_1, and the length pc_1 by b_2. By definition, we have that $a_1 + a_2 = 1$ and $b_1 + b_2 = r$, thus $a_1 + a_2 + b_1 + b_2 = 1 + r$. Because p does not lie on the geodesic Bc_2 or Ac_1, we have $a_1 + b_2 > 1$ and $a_2 + b_1 > r$ and $a_1 + a_2 + b_1 + b_2 > 1 + r$. The latter inequality yields a contradiction. □

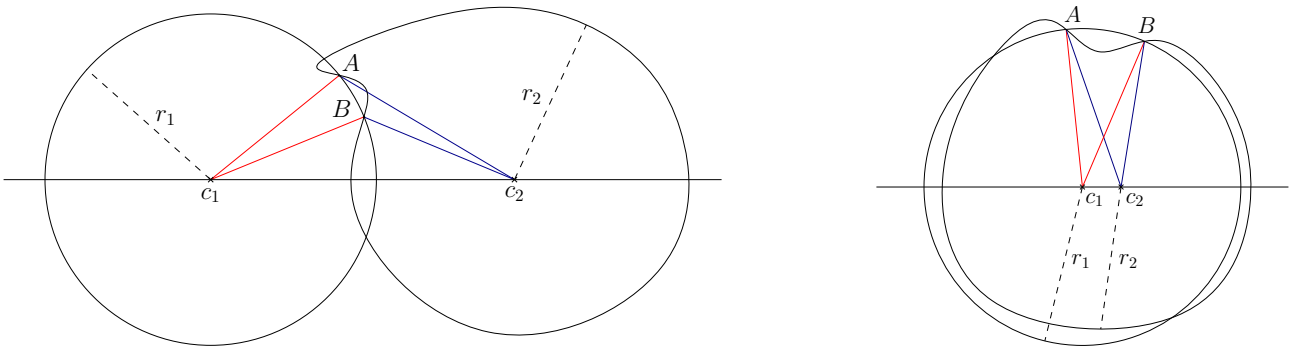

Figure 7.10: Illustrations for the proof of Lemma 7.4.8

We want to prove that the curved realization of the Delaunay complex induces a homeomorphism from the complex to the surface.

Corollary 7.4.9 *Two curved Riemannian Delaunay triangles t_1 and t_2 lying within a disk of radius less than the convexity radius on a Riemannian surface, of which no three vertices lie on a geodesic, that share an edge intersect only in this edge.*

Proof. Let v_1 and v_2 be the two vertices that are shared by $t_1 = \{v_1, v_2, v_3\}$ and $t_2 = \{v_1, v_2, v_4\}$, and denote by $\gamma_{v_1 v_2 E}$ the extended geodesic between v_1 and v_2. If v_3 and v_4 lie on either side $\gamma_{v_1 v_2 E}$, there is nothing to prove. If v_3 and v_4 lie on the same side side of $\gamma_{v_1 v_2 E}$ (say the left), we consider the geodesic circumscribing circles of t_1 and t_2, the boundaries of the disks D_{v_3} and D_{v_4}. Lemma 7.4.8 implies that these circles only intersect in v_1 and v_2. Moreover, the arc given by the part of ∂D_{v_3} on the left side of $\gamma_{v_1 v_2 E}$ is contained in D_{v_4} or vice versa the arc given by the part of ∂D_{v_4} on the left side of $\gamma_{v_1 v_2 E}$ is contained in D_{v_3}. This contradicts the empty ball property of Delaunay ball, so this configuration cannot occur. □

Corollary 7.4.10 *Let t_1 and t_2 be two curved Riemannian Delaunay triangles lying within a disk of radius less than the strong convexity radius on a Riemannian surface. Suppose that no three vertices*

lie on a geodesic, and that t_1 and t_2 share a vertex. Then the intersection of the interiors of t_1 and t_2 is empty.

Proof. Assume that this is not the case and consider the Delaunay disks for both triangles. The Delaunay disks contain the triangles because of convexity. By hypothesis, the Delaunay circles intersect in a common vertex, but since they must contain no vertices, they must intersect in at least two more points, contradicting Lemma 7.4.8. □

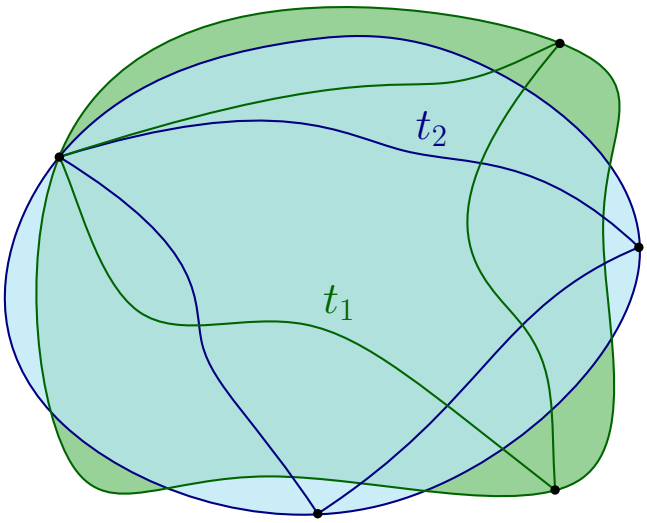

Figure 7.11: The curved Riemannian Delaunay triangles t_1 and t_2 and their respective Delaunay balls.

Remark 7.4.11 *Assume that p_i is a Delaunay vertex of a Delaunay complex, then p_i does not lie in the interior or on an edge of a Delaunay triangle, otherwise it would violate the empty ball property.*

Corollary 7.4.12 *Let p be a Delaunay vertex and denote by e_j the Delaunay edges emanating from p. Then there exists a neighborhood U around p such that all points in $U \setminus \cup_i e_i$ lie in a single Delaunay triangle.*

Proof. Because of Remark 7.4.11, there exists a neighborhood U of p such that all Delaunay triangles with empty intersection with U have p as a vertex. Corollary 7.4.10 implies that the pairwise intersections of the interiors of the simplices that have p as a vertex is empty. The statement now follows. □

Remark 7.4.13 *Suppose that all the Voronoi cells of a Voronoi diagram $\mathrm{Vor}(\{p_i\})$ are contained in a geodesic disk of radius less than a sixth of the convexity radius. Then for any p_i the Delaunay simplices that have p_i as a vertex are contained in a geodesic disk of radius less than a third of the convexity radius.*

Lemma 7.4.14 *The Voronoi cells included in a convex disk are star shaped, and in particular they are topological disks.*

Proof. Let p be the center of the Voronoi cell, and b a point on the boundary. Then the geodesic between b and p is completely contained in the Voronoi cell. If it were not the case, then there would exist a point on the geodesic between p and q that is closer to another Voronoi center q than to p. By definition of a geodesic, then b would also be closer to q. □

Lemma 7.4.15 *Let* $\mathrm{Vor}(\{p_i\})$ *be the Riemannian Voronoi diagram of the point set* $\{p_i\}$ *on an oriented Riemannian surface* $\mathcal{M}$*. Suppose that no three vertices lie on a geodesic. Assume that all Voronoi cells* V_{p_i} *lie within balls of radius less than a sixth of the convexity radius. Then every Delaunay edge is shared by at least two Delaunay triangles.*

Proof. Assume that that V_p and V_q are the Voronoi cells that share a Voronoi face. Call $\gamma_{pq,E}$ the extended geodesic between p and q in a disk centered on p of radius less than the convexity radius. The geodesic $\gamma_{pq,E}$ can be subdivided into three parts, namely the geodesic γ_{pq} between p and q and the two connected components of $\gamma_{pq,E} \setminus \gamma_{pq} = \gamma_{pq,Ep} \cup \gamma_{pq,Eq}$, with $p \in \overline{\gamma_{pq,Ep}}$ and $q \in \overline{\gamma_{pq,Eq}}$. It is clear that every point in $\gamma_{pq,Ep}$ is closer to p than q. The limit we have placed on the size of the Voronoi cell now implies that the intersection of V_p and V_q consists of segments curves. Furthermore, every Delaunay edge lies in at least one Delaunay triangle. Consider now one of the segments of the curves. Suppose that the end points of this segment consist of intersection points between the same three Voronoi cells. We can now consider the Delaunay balls centered at the end points and we find two Delaunay balls intersecting in three points, which contradicts the pseudo-ball property. Therefore, every Delaunay edge is shared by at least two Delaunay triangles. □

Lemma 7.4.16 *Under the same conditions as Lemma 7.4.15, the Delaunay complex is a topological two-manifold at every vertex.*

Proof. We consider an arbitrary vertex p_i. The vertex p_i can not be disconnected, because there exists by compactness at least one vertex p_j for which the distance between p_i and the other vertices is minimized. Therefore p_i and p_j share a Delaunay edge.

By Lemmas 7.4.15 and 7.4.9, this edge lies in two triangles whose interiors do not intersect. These triangles cannot share another edge or they would coincide. By induction over a finite (due to compactness) number of triangles, we find the star of the vertex. No overlaps can take place because of Lemma 7.4.10. □

We now have the following:

Corollary 7.4.17 *Under the same conditions as Lemma 7.4.15, the Delaunay complex is a topological two manifold.*

With this result, we are now able to state the following:

Theorem 7.4.18 *Let* $\mathcal{V}_{\{p_i\}}$ *be the Riemannian Voronoi diagram of the point set* $\{p_i\}$ *on an oriented Riemannian surface* $\mathcal{M}$*. Suppose that no three vertices lie on a geodesic. Assume that all Voronoi cells* V_{p_i} *lie within balls of radius less than a sixth of the convexity radius. Then the curved realization of the Delaunay complex induces a piecewise smooth homeomorphism from the Delaunay complex to the surface* $\mathcal{M}$*.*

Proof. We are going to apply Whitney's lemma (see Lemma 7.4.3). Because we assume that no three vertices lie on a geodesic, the curved realizations of two dimensional simplices are non-degenerate, that is diffeomorphisms. Due to Corollary 7.4.9, we can assume that two triangles that share an edge have compatible orientation. This means that the curved realization of the Delaunay complex Del induces a simplex-wise positive map, which we shall call f.

We now intersect the surface at an arbitrary point with a geodesic disk of radius $1 - \varepsilon$ times the convexity radius and remove all simplices that do not completely lie in the intersection. The resulting complex is a subcomplex of the Delaunay complex and has a boundary (which may have

more connected components and singular points). Remark 7.4.13 implies that the boundary on the surface lies at least a distance of $2/3 - \varepsilon$ times the convexity radius from the center of the geodesic disk. By Lemma 7.4.3, we now have that – with exception of the skeleton – the number of covers is constant. Applying Corollary 7.4.12, that number is one. This means that f is locally a piecewise smooth homeomorphism. Due to Remark 7.4.13 and because the point has been chosen arbitrarily, it follows that f is a global piecewise smooth homeomorphism. □

This concludes our work on curved Riemannian Delaunay triangulations in a two-dimensional setting. We now have conditions on the sampling such that the dual of the Riemannian Voronoi diagram is an embedded triangulation. Assuming that the requirements of Section 7.3 are fulfilled, the curved dual of our discrete Riemannian Voronoi diagram is a triangulation.

7.5 Straight realization of the Riemannian Voronoi diagram

We now investigate the embedding of the "straight" dual of the Riemannian Voronoi diagram. Our approach is to assume that the conditions of the previous section are satisfied – the curved dual is embedded – and to give conditions such that "straightening" the geodesic edges of the curved Riemannian Delaunay triangulation preserves the property of being a triangulation. The main step towards our goal is to show that under sufficient sampling conditions, the geodesic path and the straight edge between two seeds are close from one another (Lemma 7.5.1). We then prove that straightening edges creates no inversion of simplices or self-intersections (Theorem 7.5.3).

Lemma 7.5.1 (Distance bound between a geodesic path and a straight segment) *Let $\psi_0 \geq 1$ be a bound on the metric distortion, and g be an arbitrary metric field on U. Let U be an open neighborhood of $\Omega = \mathbb{R}^2$ that is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$ such that $\forall p \in B(p_0, r_0), \psi(g(p), g_{\mathbb{E}}(p)) \leq \psi_0$, where $g_{\mathbb{E}}$ is the Euclidean metric field.*

Let x and y be two points in the domain. and denote the geodesics with respect to g and $g_{\mathbb{E}}$ connecting x and y by γ_g and $\gamma_{\mathbb{E}}$. If $z \in \gamma_g$ then $d_{\mathbb{E}}(z, \gamma_{\mathbb{E}}) \leq a$, where

$$a = \frac{1}{2}\sqrt{\psi_0^2 - \frac{1}{\psi_0^2}}\, d_g(x, y).$$

Remark 7.5.2 *Lemma 7.5.1 and its proof could be extended painlessly to a uniform metric field g_0 instead of the Euclidean metric field $g_{\mathbb{E}}$. For this reason, we denote the Euclidean distance by $d_{\mathbb{E}}$.*

Proof. Denote by $\pi(z, \gamma_{\mathbb{E}})$ the point on $\gamma_{\mathbb{E}}$ that is closest to z. Note that since $g_{\mathbb{E}}$ is uniform, this point is unique. Additionally, if $\pi(z, \gamma_{\mathbb{E}})$ does not equal x or y, one can use Pythagoras' theorem:

$$d_{\mathbb{E}}(x, z)^2 = d_{\mathbb{E}}(x, \pi(z, \gamma_{\mathbb{E}}))^2 + d_{\mathbb{E}}(\pi(z, \gamma_{\mathbb{E}}), z)^2$$
$$d_{\mathbb{E}}(z, y)^2 = d_{\mathbb{E}}(z, \pi(z, \gamma_{\mathbb{E}}))^2 + d_{\mathbb{E}}(\pi(z, \gamma_{\mathbb{E}}), y)^2.$$

On the other hand, we have by triangular inequality that the Euclidean length $L_{\mathbb{E}}(x, z, y)$ of any curve from x to y via z is at least $d_{\mathbb{E}}(x, z) + d_{\mathbb{E}}(z, y)$.

Let $c \in \mathbb{R}$ such that $d_{\mathbb{E}}(\pi(z, \gamma_{\mathbb{E}}), x) = \frac{1}{2} d_{\mathbb{E}}(x, y) - c$ and $d_{\mathbb{E}}(\pi(z, \gamma_{\mathbb{E}}), y) = \frac{1}{2} d_{\mathbb{E}}(x, y) + c$. If $\pi(z, \gamma_{\mathbb{E}})$ does not equal x or y, then

$$L_{\mathbb{E}}(x, z, y) > \sqrt{a^2 + \left(\frac{1}{2} d_{\mathbb{E}}(x, y) - c\right)^2} + \sqrt{a^2 + \left(\frac{1}{2} d_{\mathbb{E}}(x, y) + c\right)^2}. \tag{7.5}$$

It is apparent from symmetry, but it can also easily be verified by hand that the right hand side of Inequality (7.5) is minimized with respect to c when $c = 0$, thus

$$L_{\mathbb{E}}(x, z, y) > 2\sqrt{a^2 + \left(\frac{1}{2}d_{\mathbb{E}}(x, y)\right)^2}.$$

Using the fundamental property of the metric distortion (Equation 2.1 in Chapter 2), we see that

$$L_g(x, z, y) > \frac{2}{\psi_0}\sqrt{a^2 + \left(\frac{1}{2}d_{\mathbb{E}}(x, y)\right)^2},$$

for all curves that go through z. In particular, we have assumed that $z \in \gamma_g$, therefore

$$\begin{aligned} & d_g(x, y) > \frac{2}{\psi_0}\sqrt{a^2 + \left(\frac{1}{2}d_{\mathbb{E}}(x, y)\right)^2} \\ \iff & \frac{\psi_0^2}{4}d_g(x, y)^2 > a^2 + \frac{1}{4}d_{\mathbb{E}}(x, y)^2 \geq a^2 + \frac{1}{4\psi_0^2}d_g(x, y)^2. \end{aligned}$$

Finally, bringing the metric terms to one side, we obtain

$$\frac{1}{4}\left(\psi_0^2 - \frac{1}{\psi_0^2}\right) d_g(x, y)^2 > a^2.$$

We know that if $\pi(z, \gamma_{\mathbb{E}})$ is x or y, then $L_{\mathbb{E}}(x, z, y) \geq a + \sqrt{d_{\mathbb{E}}(x, y)^2 + a^2}$. This can be easily seen as we are in Euclidean space, and $\pi(z, \gamma_{\mathbb{E}})$ is x or y, z lies on one of the two semi-circles around x and y. It now follows that $L_{\mathbb{E}}(x, z, y) \geq L_{\mathbb{E}}(x, \tilde{z}, y)$ where $\tilde{z}$ is one of the end points of the semi circle. By Pythagoras' theorem, $L_{\mathbb{E}}(x, \tilde{z}, y) = a + \sqrt{d_{\mathbb{E}}(x, y)^2 + a^2}$. □

We are ready to give the main theorem of this section.

Theorem 7.5.3 (Embedding of the straight Riemannian Delaunay complex) *If the curved dual of the Riemannian Voronoi diagram is a triangulation and if the sampling criterion ε is small enough, then the straight dual is also a triangulation.*

Proof. Consider a curved Riemannian triangle $\widetilde{\sigma}$ of the curved Riemannian Delaunay triangulation with vertices p, q and r. Notations used in this proof are illustrated in Figure 7.12.

By Lemma 3.3.1 in Chapter 3, we have bounds on the value θ_v of the angle at a Voronoi corner of the Euclidean Voronoi diagram:

$$2\arcsin\left(\frac{\mu_0}{2\varepsilon_0}\right) \leq \theta_v \leq \pi - \arcsin\left(\frac{\mu_0}{2\varepsilon_0}\right).$$

We can thus deduce bounds on the (straight) triangle angles θ_t:

$$\theta_m = \arcsin\left(\frac{\mu_0}{2\varepsilon_0}\right) \leq \theta_t \leq \pi - 2\arcsin\left(\frac{\mu_0}{2\varepsilon_0}\right) = \theta_M.$$

Using the definition of a given in Lemma 7.5.1, we have at a corner of the (straight) triangle Δpqr

$$\sin(\theta) = \frac{a}{l} \iff l = \frac{a}{\sin(\theta)}.$$

We seek a lower bound on the length of the straight edge RQ eroded by the protecting tubes of the edges pq and pr.

$$\begin{aligned} l = L - (l_R + l_Q) &\geq \mu_0 - \left(\frac{a_R}{\sin(\theta_R)} + \frac{a_Q}{\sin(\theta_Q)} \right) \\ &\geq \mu_0 - \frac{\sqrt{\psi_0^2 - \frac{1}{\psi_0^2}}\, d_{g'}(p,q)}{2\sin(\theta_R)} - \frac{\sqrt{\psi_0^2 - \frac{1}{\psi_0^2}}\, d_{g'}(p,r)}{2\sin(\theta_Q)} \\ &\geq \mu_0 - \frac{\varepsilon_0 \sqrt{\psi_0^2 - \frac{1}{\psi_0^2}}}{\sin\left(\arcsin\left(\frac{\mu_0}{2\varepsilon_0}\right)\right)} \\ &\geq \mu_0 - \frac{2\varepsilon_0^2 \sqrt{\psi_0^2 - \frac{1}{\psi_0^2}}}{\mu_0}. \end{aligned}$$

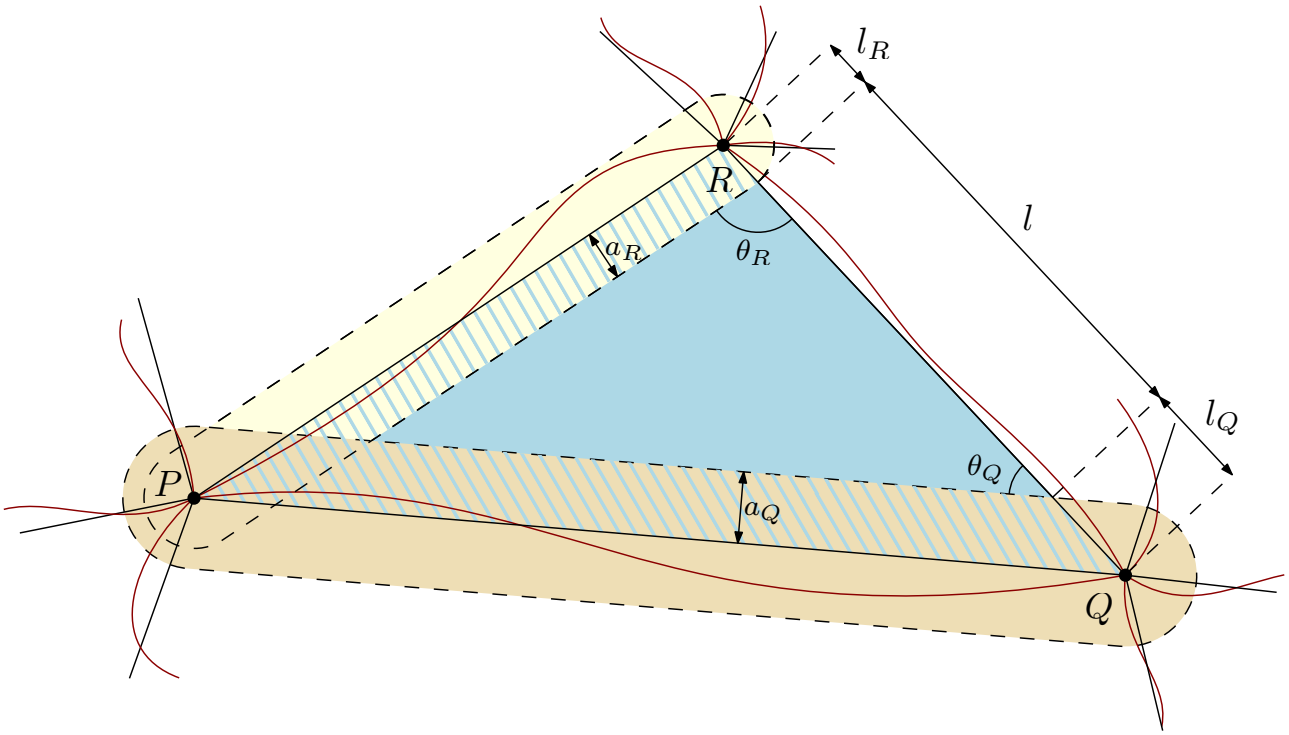

Figure 7.12: The curved Riemannian Delaunay triangulation is shown in red and the straight Riemannian Delaunay triangulation in black. The geodesic edges p-r and p-q lie in their respective protection tubes (colored in beige) around the straight edges.

If we let the metric tend to the constant metric the curved Riemannian Delaunay triangle goes to the straight Riemannian Delaunay triangle. This is a homotopy and the orientation of the triangles

remains constant if l is strictly positive. Hence, we require $l > 0$, which is verified if

$$\begin{aligned} l \geq \mu_0 - \frac{2\varepsilon_0^2\sqrt{\psi_0^2 - \frac{1}{\psi_0^2}}}{\mu_0} > 0 &\iff \mu_0^2 > 2\varepsilon_0^2\sqrt{\psi_0^2 - \frac{1}{\psi_0^2}} \\ &\iff \frac{\lambda^2}{\psi_0^2}\varepsilon^2 > 2\psi_0^2\varepsilon^2\sqrt{\psi_0^2 - \frac{1}{\psi_0^2}} \\ &\iff \lambda^2 > 2\psi_0^4\sqrt{\psi_0^2 - \frac{1}{\psi_0^2}}, \end{aligned}$$

using $\varepsilon_0 = \psi_0\varepsilon$, $\mu_0 = \mu/\psi_0$ and $\mu = \lambda\varepsilon$. The left hand side is fixed and the right hand side goes to 0 as ε goes to 0 (and ψ_0 goes to 1), proving that once the sampling is dense, no issue appears when the two edges pq and pr are straightened.

The same reasoning can then be applied to the other pairs of edges in Δpqr, giving us the existence of a point in the interior of the triangle Δpqr "eroded" by the protection zones. Straightening thus preserves the orientation of the triangle. As this can be done to all the simplices of the curved Riemannian Delaunay triangulation, the straight Riemannian Delaunay triangulation is also a triangulation of the domain. □

7.6 Construction of power protected nets

Our theory introduces requirements on the canvas $\mathcal{C}$ (through its sizing field) and on the point set $\mathcal{P}$ such that the two duals of the discrete Riemannian Voronoi diagrams are triangulations. The canvas must be sufficiently dense, and the seed set a δ-power protected (ε, μ)-net with specific values of δ and ε. The former requirement is easily achieved through refinement. A refinement algorithm to construct power protected nets was described in Section 5.4 in Chapter II.5, but it is impractical. Thankfully, neither the sampling nor the canvas need to be as dense in practice as our theory requires them to be and a farthest point refinement can be used to generate the point set.

7.7 Conclusion

We have proved in this chapter that the discrete Riemannian Voronoi diagram and the related algorithms introduced in Chapter III.6 are theoretically sound. These guarantees come in the shape of a quality requirement on the point set, through the power protected net assumption, and on the density on the canvas.

This chapter also offered a detailed approach of the reasoning that is only sketched in Chapter II.5 to prove the embeddability of the Labelle and Shewchuk anisotropic Voronoi diagram for a power protected net. Up until the involvement of the canvas, the reasoning is exactly the same with only slightly different proofs due to the non-continuous metric field that is used in the anisotropic Voronoi diagram of Labelle and Shewchuk.

It is often desirable to expose quality bounds on the simplices, which we have not presented here. Additionally, it would be interesting to estimate the loss of quality that results from straightening Riemannian simplices into straight simplices: we have seen in practice that this can be quite significant (see Chapter III.6).

In the next chapter, we shall investigate the theoretical soundness of our algorithm with the assumption of a domain of arbitrary dimension. Most of the proofs that we have used in this chapter do not extend to that setting or are not sufficient. For example, proving that the geodesic edges and

straight edges are close is not sufficient to prove that there is no inversion when straightening a curved Riemannian Delaunay triangulation into a straight Delaunay triangulation. We shall introduce more complex notions, but worse bounds are often obtained.

8. Theoretical study of n-dimensional discrete Riemannian Voronoi diagrams

This chapter is devoted to the theoretical study of the discrete Riemannian Voronoi diagram, introduced in Chapter III.6, in the setting of a manifold of arbitrary dimension. We have studied in Chapter III.7 the theoretical aspect of the structure in the context of two-dimensional manifolds and exposed sufficient conditions on the quality of the point set and the density of the canvas such that the duals of our discrete Riemannian Voronoi diagram using both curved and straight Riemannian simplices simplices are triangulations. Unfortunately, the proofs of Chapter III.7 do not extend to higher dimensions.

In this chapter, we extend the results of Chapter III.7 to the setting of manifolds of arbitrary dimension. The skeleton of this chapter is similar into three parts. We first obtain the combinatorial equivalence between the nerves of the discrete Riemannian Voronoi diagram and the Riemannian Voronoi diagram. Contrary to the previous chapter, we do not introduce any new result on the embeddability of the curved Riemannian Delaunay complex (the dual the Riemannian Voronoi diagram using curved Riemannian simplices) and fully rely on existing work, which is sufficient to obtain conditions this embeddability. Finally, we prove that while "straightening" curved Riemannian simplices to obtain a classical straight Delaunay triangulation we do not lose the embedding of the complex.

As both Chapters III.7 and III.8 seeks to prove identical results – albeit with a large difference in the assumption of the dimension – and follow a similar approach, it might seem strange to split these chapters, or to even detail the two-dimensional case. However, Chapter III.7 has the benefit of much simpler proofs and better bounds, thus offering a clear presentation of our approach. Additionally, the overlap in the goals and the global approach is not carried over to the results and the actual proofs.

Contributions In this chapter, we extend the theoretical study of our algorithm from the setting of 2-manifolds to any dimension. This is not straightforward and often results in an increased complexity and worse bounds over the two-dimensional setting. A number of intermediary results are very technical and were presented in Chapter 3 for clarity.

A number of results are direct generalizations of results presented in the two-dimensional setting in Chapter III.7. These however sometimes require significantly different proofs. For example, the embedding of the straight Delaunay triangulation is also achieved by bounding the distance between a Riemannian and straight facet of a simplex, but the approach is unlike that of Chapter III.7. The proof of the combinatorial equivalence between the nerves of the discrete Riemannian Voronoi diagram and the Riemannian Voronoi diagram in a flat domain endowed with an Euclidean metric field is the most significant change and is achieved with an elegant coloring result by Sperner.

Contents

8.1 Background

We begin by recalling notations introduced in the previous chapter that shall be employed in this chapter. Details can be found in Chapter III.6.

We consider a set of seeds $\mathcal{P}$ in a domain Ω, and each seed is attributed a color. The domain is endowed with an arbitrary metric field g, and the geodesic distance between two points, denoted by d_g, is the length of the shortest path between these two points. The Voronoi diagram constructed with the geodesic distance is the Riemannian Voronoi diagram $\mathrm{Vor}_g(\mathcal{P})$. As this diagram is difficult to compute in practice, we approximate the geodesic distance and the diagram $\mathrm{Vor}_g(\mathcal{P})$ by considering a discrete underlying structure, the canvas $\mathcal{C}$, whose vertices are colored according to their closest seed. This coloring is obtained through a multiple-front Dijkstra algorithm from the seeds. The discrete Voronoi cell of a seed is the set of simplices that possess at least one vertex colored with the color of the seed. The discrete Riemannian Voronoi diagram $\mathrm{Vor}_g^d(\mathcal{P})$ is the set of discrete Voronoi cells. An example of such a diagram is illustrated on Figure 8.1. The nerve of a Voronoi diagram $\mathrm{Vor}_d(\mathcal{P})$ is a formal way to express its combinatorial structure and is defined by

$$\mathcal{N}(\mathrm{Vor}_d(\mathcal{P})) = \{\{\mathrm{V}_d(p_i)\} \mid \cap_i \mathrm{V}_d(p_i) \neq \emptyset, p_i \in \mathcal{P}\}.$$

The straight and curved duals of a Voronoi diagram are realizations of the nerve with vertices in $\mathcal{P}$ and using respectively straight and curved Riemannian Delaunay simplices (see Sections 2.7 and 2.8 in Chapter 2 for additional details on the definition). If a dual is embedded in the domain, we say that it is a triangulation.

8.2 Overview

As is the case for most Voronoi diagrams, the (Riemannian or straight) duals of a discrete Riemannian Voronoi diagram is generally not an embedded complex (triangulation). To prove the embeddability of the duals of the discrete Voronoi diagram, we follow the same indirect path as in the previous chapter: we first expose conditions such that the discrete Riemannian Voronoi diagram and the Riemannian Voronoi diagram have the same nerve (combinatorial structure). We then expose

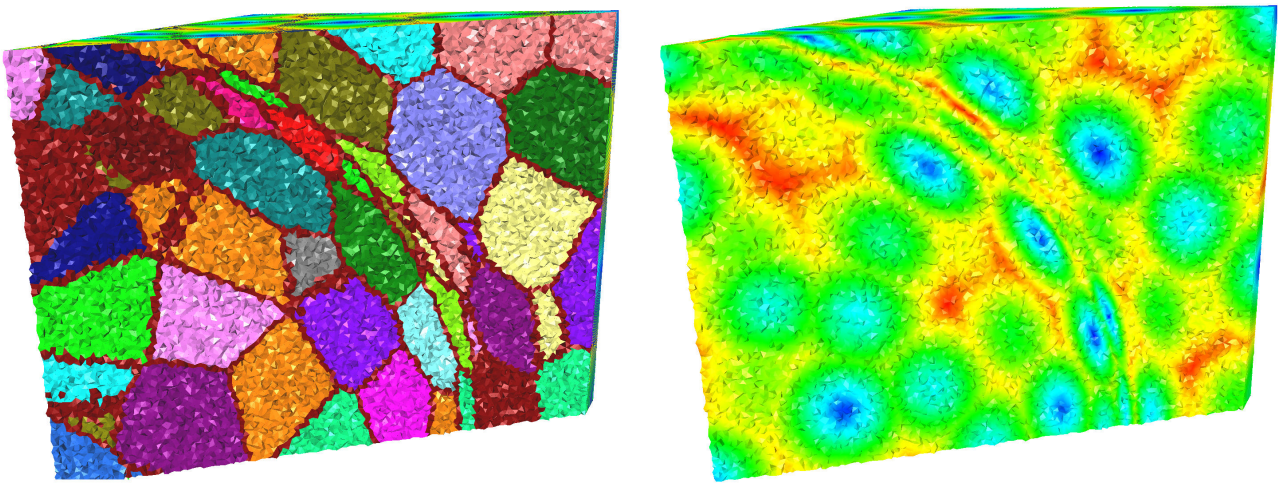

Figure 8.1: Slice of a three-dimensional discrete Riemannian Voronoi diagram over an isotropic canvas. The discrete cells are drawn on the left, and the geodesic distance maps on the right.

conditions such that the curved dual of the discrete Riemannian Voronoi diagram is a triangulation. Finally, we straighten the curved Riemannian simplices of the curved dual and show that – under sufficient conditions – no inversion can appear in the triangulation, thus preserving an embedded dual.

We now give an overview of the approaches followed and the theoretical results achieved in this chapter.

8.2.1 Nerves of the discrete and Riemannian Voronoi diagrams

Although the discrete Riemannian Voronoi diagram by definition only relies on discrete information, we show that its combinatorial structure can be precisely that of the Riemannian Voronoi diagram. When this is the case, we say that the nerves of the diagrams are *equivalent* and write $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}_g(\mathcal{P}))$.

Our approach is largely identical to that of Chapter III.7; we seek to satisfy the same two sufficient conditions that, when combined, give the equivalence, namely:

(1) for any Voronoi vertex v in the Riemannian Voronoi diagram, there is at least one canvas simplex that possesses the colors of the Voronoi cells incident to v;

(2) no canvas simplex witnesses combinatorial information that does not belong to the nerve (that is, no canvas simplex has vertices whose colors are those of non-adjacent Riemannian Voronoi cells).

As in Chapter III.7, requirements are necessary on both the nature of the set of seeds and the density of the canvas to guarantee that $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}_g(\mathcal{P}))$. A recurring theme throughout this thesis, power protected nets (defined in Sections 2.12.1 and 2.12.2 of Chapter 2) will once again show their strength in these proofs. We discuss the generation of such seed sets in Section 8.6.

Approach

The equivalence of the nerves for a generic manifold of arbitrary dimension endowed with a generic Lipschitz-continuous metric field is shown in Section 8.3.4 and specifically in Theorem 8.3.11.

Our analysis is divided in four parts. We first prove the equivalence at the most basic case of a flat domain endowed with the Euclidean metric field. The conditions exposed in the basic setting can be transported to the setting of a uniform metric field by using the stretching transformation given by the square root of a metric. An arbitrary Riemannian metric field is then considered over a flat domain, and the fact that a metric field can be locally well approximated by a uniform metric field is used to extend the results from the previous cases. Finally, we solve the general setting by locally approximating the manifold at a point with its tangent plane.

Sperner's lemma One of the most significant changes from Chapter III.7 concerns the manner in which the combinatorial equivalence of the nerves is achieved in the setting of a flat domain endowed with an Euclidean metric field significantly differs from the two-dimensional setting. To obtain $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}_g(\mathcal{P}))$, it is clearly sufficient to show that Conditions (1) and (2) are satisfied. The latter condition is easily satisfied once a bound is known on the distance between Voronoi vertices, but the former requires more work. In the two-dimensional setting, we have proved that, starting from the canvas simplex that contains a Voronoi vertex, walk on the canvas and eventually find a canvas simplex that would witness the Voronoi vertex. This method does not extend to higher dimensions, and we employ a completely different approach.

We use Sperner's lemma [131], a classical result of combinatorics, which can be seen as the discrete analog of Brouwer's fixed point theorem and gives the existence of a simplex that is colored with a complete set of colors within a triangulation, provided that this triangulation fulfills a set of topological and color-related assumptions. We shall apply this lemma to our canvas $\mathcal{C}$ and show that for every Voronoi vertex v of the Riemannian Voronoi diagram, we can construct from a subset $\mathcal{C}_v$ of $\mathcal{C}$ a (triangulated) simplex that fulfills the assumptions of Sperner's lemma, thus obtaining the existence of a canvas simplex that witnesses v.

8.2.2 Embeddability of the curved Riemannian Delaunay triangulations

Once it is proven that our discrete diagram has the same combinatorial structure as the Riemannian Voronoi diagram, we build upon known theoretical results on the curved Riemannian Delaunay triangulation to show that the curved dual of a discrete Riemannian Voronoi diagram can be – under sufficient conditions – embedded as a triangulation of the point set using curved Riemannian simplices. For this part, we entirely rely on previous results on the embeddability of curved Riemannian Delaunay triangulations. The main result is given by Theorem 8.4.4.

8.2.3 Embeddability of the straight Riemannian Delaunay triangulations

The last part of this chapter gives conditions such that the curved Riemannian simplices of the curved Riemannian Delaunay triangulation can be straightened to form the straight dual of the discrete Riemannian Voronoi diagram and that this dual is also embedded. Here, the key intermediary result is the computation of a bound on the distance between a point on a Riemannian facet and its equivalent on the straightened facet. This bound depends on the quality of the set of seeds and on the local distortion of the metric field. We then prove that when this point to point distance is sufficiently bounded, no inversion can happen while deforming from a curved Riemannian simplex to a straight simplex.

8.2.4 Nature of the canvas

To simplify computations, we assume (as in Chapter III.7) that the canvas is an isotropic Delaunay triangulation of the domain. This triangulation is further assumed to be generated through a Delaunay refinement algorithm driven by a uniform sizing field $h_{\mathcal{C}}$ that bounds the circumradius of Delaunay simplices. This assumption does not reduce the genericity of the approach: these proofs can be easily extended to star set canvasses as a star set canvas eventually becomes a consistent triangulation when well-chosen vertices are inserted (see Chapter I.4).

8.3 Equivalence of the nerves of the discrete and Riemannian Voronoi diagrams

This section is devoted to proving the combinatorial equivalence between the nerves of the discrete and Riemannian Voronoi diagrams: under sufficient conditions on the density of the canvas, we have $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}_g(\mathcal{P}))$.

To simplify matters, we initially assume that geodesic distances are computed exactly on the canvas.

8.3.1 Euclidean metric field

We first consider the simplest setting of a flat domain of dimension n endowed with the Euclidean metric field. The following theorem gives the sufficient conditions that we seek.

Theorem 8.3.1 *Assume that $\mathcal{P}$ is a δ-power protected (ε, μ)-net of $\mathbb{R}^n$ with respect to the Euclidean metric field $g_{\mathbb{E}}$. Denote by $\mathcal{C}$ the canvas, a Delaunay triangulation of $\mathbb{R}^n$ with sizing field $h_{\mathcal{C}}$. If*

$$h_{\mathcal{C}} < \min\left\{\frac{\mu}{32}, \frac{\delta^2}{128\varepsilon}\right\},$$

then $\mathcal{N}(\mathrm{Vor}_{\mathbb{E}}^d(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}_{\mathbb{E}}^d(\mathcal{P}))$.

We shall prove this theorem by enforcing Conditions (1) and (2). Rephrasing the first condition, we seek requirements on the density of the canvas $\mathcal{C}$ and the nature of the point set such that there exists at least one (maximal) canvas simplex of $\mathcal{C}$ that has exactly the colors $c_0, \ldots, c_d$ of the cells in $D = \{\mathrm{V}_g(p_i)\}$, for all $D \in \mathcal{N}(\mathrm{Vor}_{\mathbb{E}}(\mathcal{P}))$ with $|D| = n + 1$.

As explained in the overview, the core of our proof is a combinatorial result by Sperner [131], which we recall below and illustrate on Figure 8.2.

Theorem 8.3.2 (Sperner's lemma) *Let $\sigma = (e_0, \ldots, e_n)$ be an n-simplex and let T_σ denote a triangulation of the simplex. Let each vertex $e' \in T_\sigma$ be colored such that the following conditions are satisfied:*

- *The vertices e_i of σ all have different colors.*
- *If e' lies on a k-face $(e_{i_0}, \ldots e_{i_k})$ of σ, then e' has the same color as one of the vertices of the face, that is e_{i_j}.*

Then, there exists an odd number of simplices in T_σ whose vertices are colored with all $n + 1$ colors. In particular, there must be at least one.

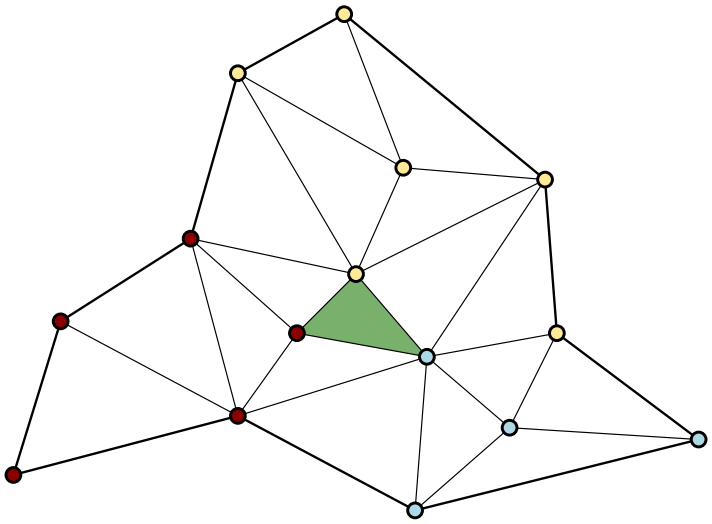

Figure 8.2: Illustration of Sperner's lemma in 2D. The green triangle is colored with all 3 colors.

We will apply this lemma to our canvas $\mathcal{C}$ and show that for every Voronoi vertex v of the Riemannian Voronoi diagram, we can find a subset $\mathcal{C}_v$ of $\mathcal{C}$ that is a triangulation of a simplex that fulfills the assumptions of Sperner's lemma, thus obtaining the existence of a canvas simplex (in $\mathcal{C}_v$ and thus in $\mathcal{C}$) that witnesses the Voronoi vertex.

The construction of $\mathcal{C}_v$ is detailed in the following sections and illustrated in Figure 8.3: starting from a colored canvas (left), we subdivide σ_v, the dual of v, to obtain $\mathcal{T}_v$ (middle). We then deduce a set of simplices $\mathcal{C}_v$ whose central elements are canvas simplices. $\mathcal{C}_v$ is shown to form the triangulation of a simplex that satisfies the hypotheses of Sperner's lemma, giving the existence of a canvas simplex (in green, right) that witnesses the Voronoi vertex within the union of the simplices, and therefore in the canvas.

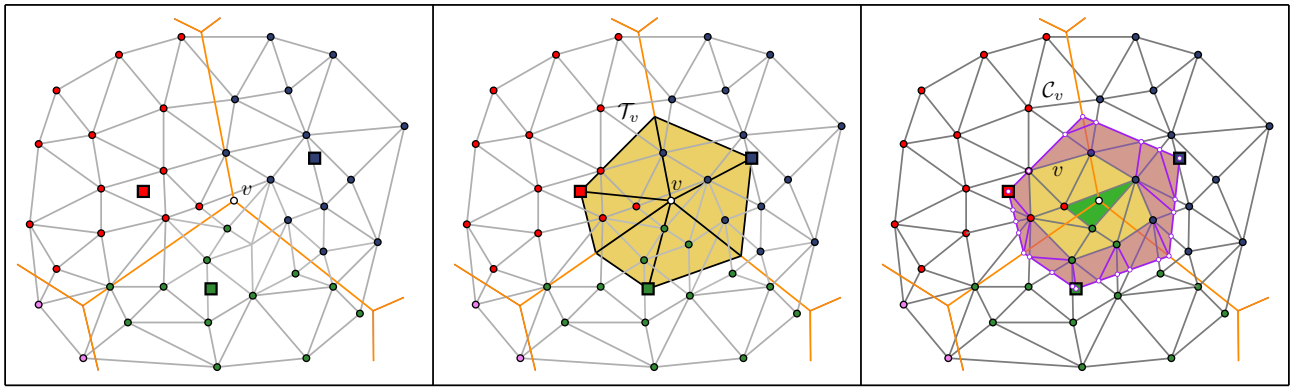

Figure 8.3: Illustration of the construction of $\mathcal{C}_v$. The Riemannian Voronoi diagram is drawn with thick orange edges and the sites are colored squares. The canvas is drawn with thin gray edges and colored circular vertices. The middle frame shows the subdivision of the dual simplex σ_v with think black edges and the triangulation $\mathcal{T}_v$ is drawn in yellow. On the right frame, the set of simplices $\mathcal{C}_v$ is colored in purple (simplices that do not belong to $\mathcal{C}$) and in dark yellow (simplices that belong to $\mathcal{C}$).

Building the triangulation $\mathcal{T}_v$

Consider the simplex σ_v dual of v. As explained in the introduction of this section, it is not possible to ensure that a face τ_v of σ_v lives in the union of the Voronoi cells of the vertices of τ_v. Instead, we apply a barycentric subdivision to σ_v (see Figure 8.4).

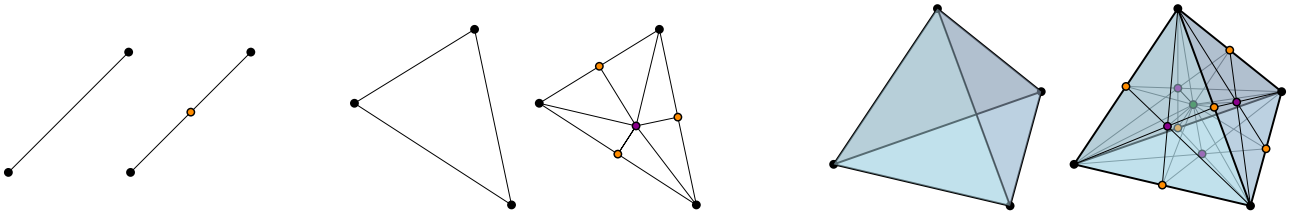

Figure 8.4: Illustration of the barycentric subdivision of an edge, a triangle and a tetrahedron.

For a given Voronoi vertex v in the Euclidean Voronoi diagram $\mathrm{Vor}_{\mathbb{E}}(\mathcal{P})$ of the domain Ω, the initial triangulation $\mathcal{T}_v$ is obtained by applying a combinatorial barycentric subdivision of simplex σ_v. To each face τ_v of σ_v with dual F, we associate a point c_F in F which is not necessarily the geometric barycenter. We randomly associate to c_F the color of any of the sites whose Voronoi cells intersect to give F. For example, in a two-dimensional setting, if τ_v is an edge then the face F is a Voronoi edge that is the intersection of V_{red} and V_{blue}, and c_F is colored either red or blue.

Figure 8.5 illustrates the process of breaking down a facet of a tetrahedron through barycentric subdivision: the facet prs is broken down in 6 green triangles that all live within the union of the (Euclidean) Voronoi cells $\mathrm{V}(p)$, $\mathrm{V}(r)$ and $\mathrm{V}(s)$.

Remark 8.3.3 *If τ_v is a 0-face, then F is a Voronoi cell. In that specific case, c_F is not random, but the site of the Voronoi cell is chosen as c_F.*

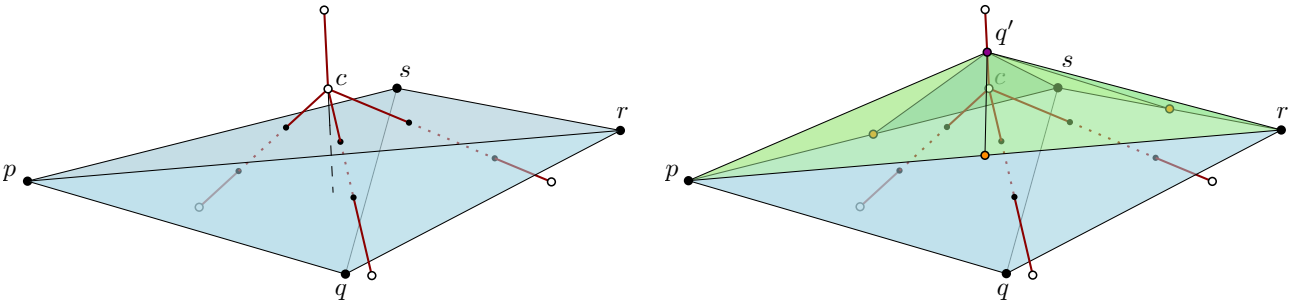

Figure 8.5: Subdivision of the facet prs of the tetrahedron $pqrs$ (blue) into 6 triangles (green). Voronoi vertices are indicated with white circles and Voronoi edges are drawn in red.

Then, the subdivision of σ_v is computed by associating to all possible sequences of Voronoi faces $\{F_0, F_1, \dots F_{n-1}, F_n\}$ such that $F_0 \subset F_1 \cdots \subset F_n = V$ and $\dim(F_{i+1}) = \dim(F_i) + 1$ the simplex with vertices $\{c_{F_0}, c_{F_1}, \dots, c_{F_{n-1}}, c_{F_n}\}$. These barycentric subdivisions are allowed since Voronoi cells are convex polytopes. The triangulation $\mathcal{T}_v$ is illustrated in Figure 8.6 in dimension 3.

As shall be proven in Lemma 8.3.5, $\mathcal{T}_v$ can be used to define a combinatorial simplex that satisfies the assumptions of Sperner's lemma.

Remark 8.3.4 *Choosing a Voronoi $(k+1)$-face F_{k+1} that contains a Voronoi k-face F_k is simply removing a vertex from the set $\mathcal{P}_{|k}$ of seeds such that $F_k = q(\cap_{p_i \in \mathcal{P}_{|k}} V(p_i))$. Since F_k is of dimension*

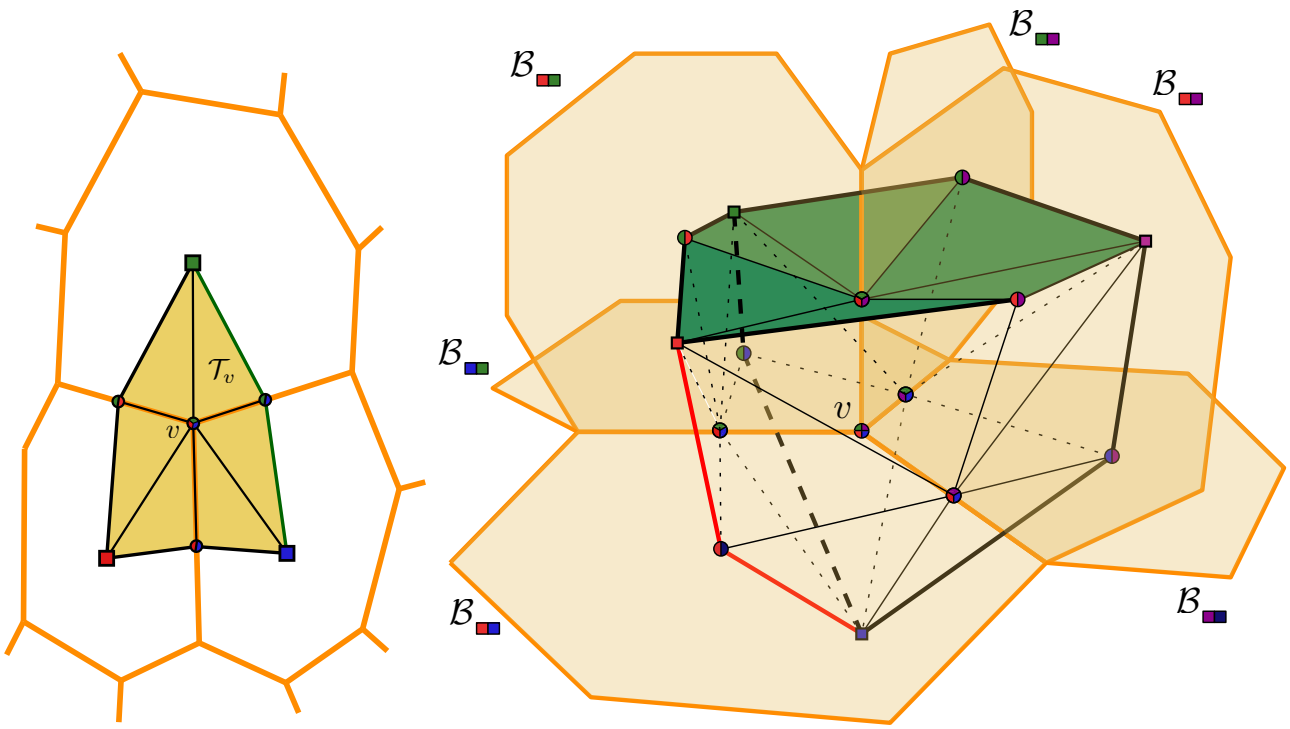

Figure 8.6: Illustration of the triangulations $\mathcal{T}_v$ in 2D and 3D. The bisectors of the Riemannian Voronoi diagram are colored in orange. Segments hidden behind Voronoi faces are either dashed or dotted. Seeds are colored squares. On the right, a face (in green) and an edge (in red) of $\sigma_{\mathcal{S}}$.

k, we have $\left|\mathcal{P}_{|k}\right| = (n+1)-k$. There are thus $(n+1)-k$ $(k+1)$-faces that contain F_k. The cardinality of $\Sigma_{v,0} = \{\mathcal{F}, |\mathcal{P}| = n+1\}$ can be evaluated:

$$|\Sigma_{v,0}| = \prod_{i=0}^{n}(n+1) - i = \prod_{i=1}^{n+1} i = (n+1)\,!.$$

Observe that this is indeed the case in Figure 8.6: in 2D, $\Sigma_{v,0}$ has 6 triangles, and, in 3D, $\Sigma_{v,0}$ has 24 tetrahedra (there are 6 triangles per combinatorial face and each triangle makes a tetrahedron with c).

$\mathcal{T}_v$ as a triangulation of an n-simplex By construction, the triangulation $\mathcal{T}_v$ is a triangulation of the (Euclidean) Delaunay simplex σ_v dual of v as follows. We first perform the standard barycentric subdivision on this Delaunay simplex σ_v. We then map the barycenter of a k-face τ_v of σ_v to the point c_{F_i} on the Voronoi face F_i, where F_i is the Voronoi dual of the k-face τ_v. This gives a piecewise linear homeomorphism from the Delaunay simplex σ_v to the triangulation $\mathcal{T}_v$. We call the image of this map the simplex $\sigma_{\mathcal{S}}$ and refer to the images of the faces of the Delaunay simplex as the faces of $\sigma_{\mathcal{S}}$. We can now apply Sperner's lemma.

Lemma 8.3.5 *Let $\mathcal{P}$ be a point set. Let v be a Voronoi vertex in the Euclidean Voronoi diagram, $\mathrm{Vor}_{\mathbb{E}}(\mathcal{P})$, and let Σ_v be defined as above. The simplex $\sigma_{\mathcal{S}}$ and the triangulation $\mathcal{T}_v$ satisfy the assumptions of Sperner's lemma in dimension n.*

Proof. By the piecewise linear map that we have described above, $\mathcal{T}_v$ is a triangulation of the simplex $\sigma_{\mathcal{S}}$. Because by construction the vertices c_{F_i} lie on the Voronoi duals F_i of the corresponding

Delaunay face τ_v, c_{F_i} has the one of the colors of of the Delaunay vertices of τ_v. Therefore, $\sigma_{\mathcal{S}}$ satisfies the assumptions of Sperner's lemma and there exists an n-simplex in $\mathcal{T}_v$ that witnesses v and its dual simplex σ_v. □

Building the triangulation $\mathcal{C}_v$

Let p_i be the vertices of the k-face $\tau_{\mathcal{S}}$ of $\sigma_{\mathcal{S}}$. In this section we shall assume not only that $\tau_{\mathcal{S}}$ is contained in the union of the Voronoi cells of $V(p_i)$, but in fact that $\tau_{\mathcal{S}}$ is a distance $8e_{\mathcal{C}}$ removed from the boundary of $\cup V(p_i)$, where $e_{\mathcal{C}}$ is the longest edge length of a simplex in the canvas.

We will now construct a triangulation $\mathcal{C}_v$ of $\sigma_{\mathcal{S}}$ such that:

- $\sigma_{\mathcal{S}}$ and its triangulation $\mathcal{C}_v$ satisfies the conditions of Sperner's lemma,
- the simplices of $\mathcal{C}_v$ that have no vertex that lies on the boundary $\partial\sigma_{\mathcal{S}}$ are simplices of the canvas $\mathcal{C}$.

The construction goes as follows: We first intersect the canvas $\mathcal{C}$ with $\sigma_{\mathcal{S}}$ and consider the canvas simplices $\sigma_{\mathcal{C},i}$ such that the intersection of $\sigma_{\mathcal{S}}$ and $\sigma_{\mathcal{C},i}$ in non-empty. These simplices $\sigma_{\mathcal{C},i}$ can be subdivided into two sets, namely those that lie entirely in the interior of $\sigma_{\mathcal{S}}$, which we denote by $\sigma^{\text{int}}_{\mathcal{C},i}$, and those that intersect the boundary, denoted by $\sigma^{\partial}_{\mathcal{C},i}$.

The simplices $\sigma^{\text{int}}_{\mathcal{C},i}$ are added to the set $\mathcal{C}_v$. We intersect the simplices $\sigma^{\partial}_{\mathcal{C},i}$ with $\sigma_{\mathcal{S}}$ and triangulate the intersection. Note that $\sigma^{\partial}_{\mathcal{C},i} \cap \sigma_{\mathcal{S}}$ is a convex polyhedron and thus triangulating it is not a difficult task. The vertices of the simplices in the triangulation of $\sigma^{\partial}_{\mathcal{C},i} \cap \sigma_{\mathcal{S}}$ are colored according to which Voronoi cell they belong to. Finally, the simplices in the triangulation of $\sigma^{\partial}_{\mathcal{C},i} \cap \sigma_{\mathcal{S}}$ are added to the set $\mathcal{C}_v$.

Since $\mathcal{T}_v$ is a triangulation of $\sigma_{\mathcal{S}}$, the set $\mathcal{C}_v$ is by construction also a triangulation of $\sigma_{\mathcal{S}}$. This triangulation trivially gives a triangulation of the faces $\tau_{\mathcal{S}}$. Because we assume that $\tau_{\mathcal{S}}$ is contained in the union of its Voronoi cells, with a margin of $8e_{\mathcal{C}}$ we now can draw two important conclusions:

- The vertices of the triangulation of each face $\tau_{\mathcal{S}}$ have the colors of the vertices p_i of $\tau_{\mathcal{S}}$.
- None of the simplices in the triangulation of $\sigma^{\partial}_{\mathcal{C},i} \cap \sigma_{\mathcal{S}}$ can have $n+1$ colors, because every such simplex must be close to one face $\tau_{\mathcal{S}}$, which means that it must be contained in the union of the Voronoi cells $V(p_i)$ of the vertices of $\tau_{\mathcal{S}}$.

We can now invoke Sperner's lemma; $\mathcal{C}_v$ is a triangulation of the simplex $\sigma_{\mathcal{S}}$ whose every face has been colored with the appropriate colors (since $\sigma_{\mathcal{S}}$ triangulated by $\mathcal{T}_v$ satisfies the assumptions of Sperner's lemma, see Lemma 8.3.5). This means that there is a simplex $\mathcal{C}_v$ that is colored with $n+1$ colors. Because of our second observation above, the simplex with these $n+1$ colors must lie in the interior of $\sigma_{\mathcal{S}}$ and is thus a canvas simplex.

We summarize by the following lemma:

Lemma 8.3.6 *If every face $\tau_{\mathcal{S}}$ of $\sigma_{\mathcal{S}}$ with vertices p_i is at distance $8e_{\mathcal{C}}$ from the boundary of the union of its Voronoi cells $\partial(\cup V(p_i))$, then there exists a canvas simplex in $\mathcal{C}_v$ such that it is colored with the same vertices as the vertices of $\sigma_{\mathcal{S}}$.*

The key task that we have now is to guarantee that faces $\tau_{\mathcal{S}}$ indeed lie well inside of the union of the appropriate Voronoi regions. This requires first and foremost power protection. Indeed, if a point set is power protected, the distance between a Voronoi vertex c and the Voronoi faces that are not incident to c, which we will refer to from now on as *foreign* Voronoi faces, can be bounded, as shown in the following Lemma:

Lemma 8.3.7 *Suppose that c is the circumcenter of a δ-power protected simplex σ of a Delaunay triangulation built from an ε-sample, then all foreign Voronoi faces are at least $\delta^2/8\varepsilon$ far from c.*

This lemma is proved in Section 3.2.2 of Chapter 3.

In almost all cases, this result gives us the distance bound we require: we can assume that vertices $\{c_{F_0}, c_{F_1}, \ldots, c_{F_{n-1}}, c_{F_n}\}$ on the Voronoi faces, which we used to construct $\mathcal{T}_v$, are well placed, meaning there is a minimum distance between these vertices and foreign Voronoi objects.

However it can still occur that foreign Voronoi entities are close to a face $\tau_{\mathcal{S}}$ of $\sigma_{\mathcal{S}}$. This occurs even in two dimensions where a Voronoi vertex v' can be very close to a face $\tau_{\mathcal{S}}$ because of obtuse angles, as illustrated in Figures 8.7 and 8.8.

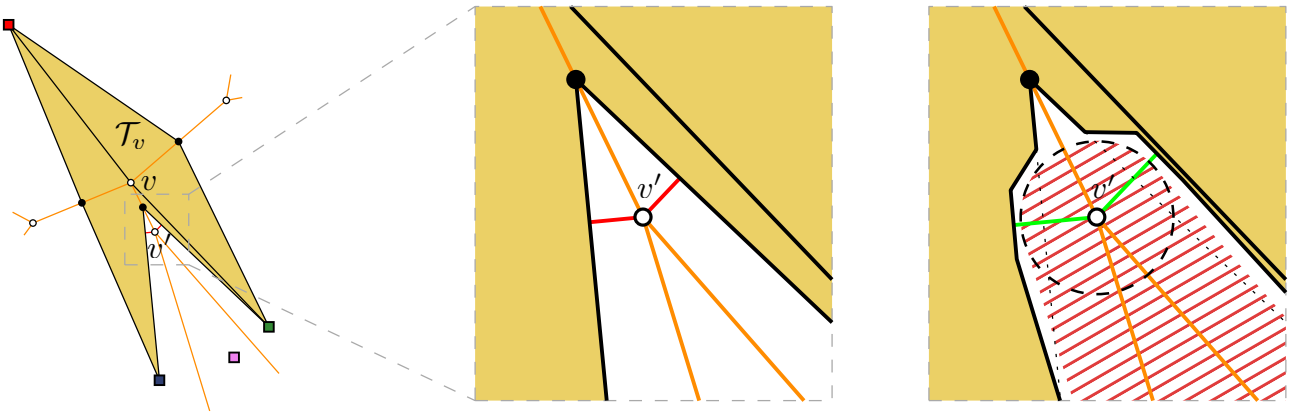

Figure 8.7: The point v' can be arbitrarily close to $\mathcal{T}_v$, as shown by the red segments (left and center). After piecewise linear deformation, this issue is resolved, as seen by the green segments (right).

Because of power protection we know that v' is removed from foreign Voronoi objects. We can therefore deform $\sigma_{\mathcal{S}}$ (in a piecewise linear manner) in a neighborhood of v' such that the distance between v' and all the faces of the deformed $\sigma_{\mathcal{S}}$ is lower bounded.

In general, the deformation of $\sigma_{\mathcal{S}}$ is performed by "radially pushing" simplices away from the foreign Voronoi faces of v with a ball of radius $r = \min\{\mu/16, \delta^2/64\varepsilon\}$. The value $\mu/16$ is chosen so that we do not move any vertex of σ_v (the dual of v): indeed, $\mathcal{P}$ is μ-separated and thus $d_{\mathbb{E}}(p_i, p_j) > \mu$. The value $\delta^2/64\varepsilon$ is chosen so that $\sigma_{\mathcal{S}}$ and its deformation stay isotopic (no "pinching" can happen), using Lemma 8.3.7. In fact it is advisable to use a piecewise linear version of "radial pushing", to ensure that the deformation of $\sigma_{\mathcal{S}}$ is a polyhedron. This guarantees that we can triangulate the intersection, see Chapter 2 of Rourke and Sanderson [121]. After this deformation we can follow the steps that we have given above to arrive at a well-colored simplex.

Lemma 8.3.8 *Let $\mathcal{P}$ be a δ-power protected (ε, μ)-net. Let v be a Voronoi vertex of the Euclidean Voronoi diagram $\mathrm{Vor}_{\mathbb{E}}(\mathcal{P})$, and $\mathcal{T}_v$ as defined above. If the length $e_{\mathcal{C}}$ of the longest canvas edge is bounded as follows: $e_{\mathcal{C}} < r = \min\{\mu/16, \delta^2/64\varepsilon\}$, then there exists a canvas simplex that witnesses v and the corresponding simplex σ_v in $\mathrm{Del}_{\mathbb{E}}(\mathcal{P})$.*

Extreme deformation of faces An extreme configuration can have a sphere in Figure 8.8 separating entirely a zone from the Voronoi vertex. In that case, we can chose different spheres to "push away" faces. Figure 8.9 shows the construction. We do not detail the computations.

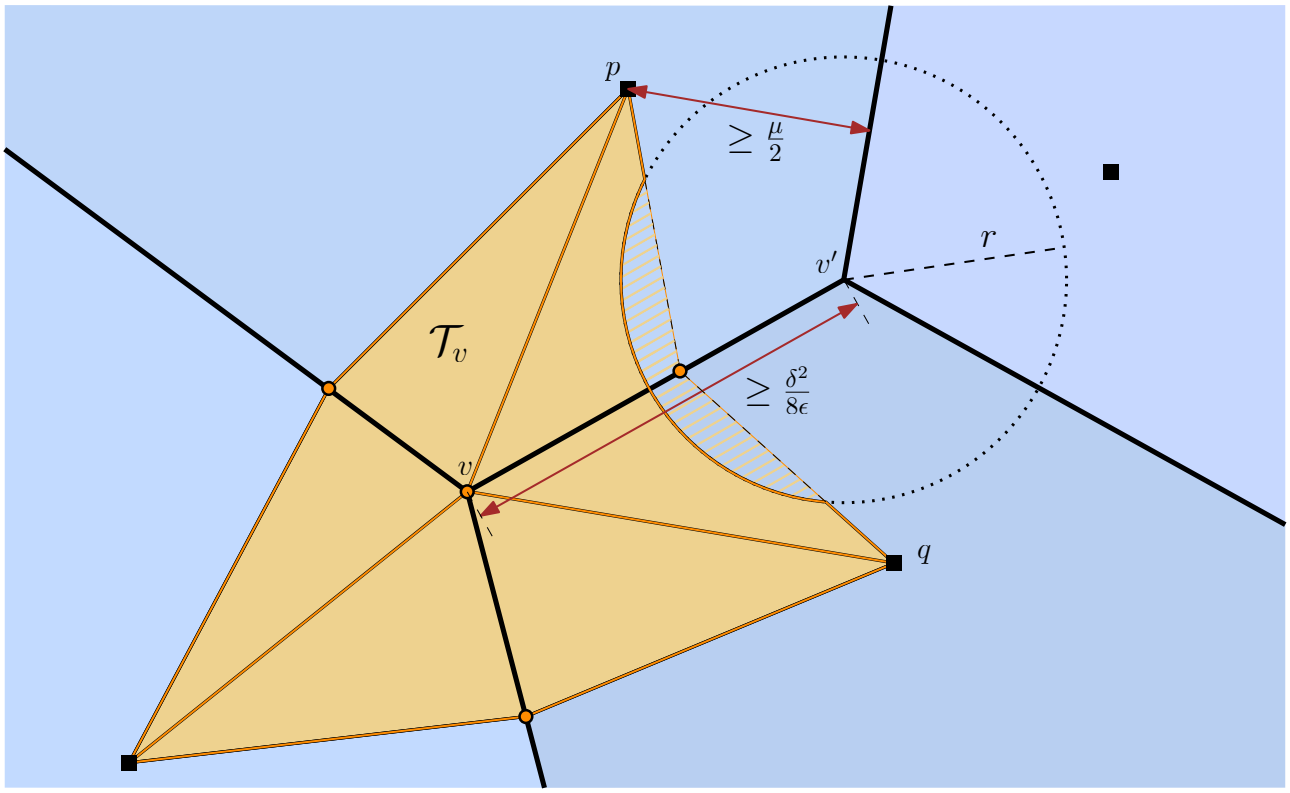

Figure 8.8: Deformation of $\mathcal{T}_v$ that clips a c_F (this does not create any issue). Dashed parts show $\mathcal{T}_v$ before deformation.

Conclusion So far, we have only proven that Condition (1), that is if there exists $D \in \mathcal{N}(\mathrm{Vor}_g(\mathcal{P}))$, then there exists D' in $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P}))$ with the same combinatorial information. The other inclusion, which corresponds to Condition (2) mentioned above, is much simpler: as long as a canvas edge is shorter than the smallest distance between a Voronoi vertex and a foreign face of the Riemannian Voronoi diagram, then no canvas simplex can witness a simplex that is not in $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P}))$. Such a bound is already given by Lemma 8.3.7 and thus, if $h_{\mathcal{C}} < \delta^2/16\varepsilon$ then we have the other inclusion. Observe that this requirement is weaker than the condition imposed in Lemma 8.3.8 and it was thus already satisfied. It follows that $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}_g(\mathcal{P}))$ if $h_{\mathcal{C}} < \min\{\mu/32, \delta^2/128\varepsilon\}$, which concludes the proof of Theorem 8.3.1.

This concludes the proof of Theorem 8.3.1.

Remark 8.3.9 *As expected, a low separation criterion μ or a low power protection criterion δ must be compensated with a smaller sizing field for the canvas to ensure that the nerves are equivalent.*

8.3.2 Uniform metric field

We now investigate the setting of a flat domain endowed with a uniform metric field.

By Lemma 3.4.1 in Chapter 3, if a point set $\mathcal{P}$ in $\mathbb{R}^n$ is a δ-power protected (ε, μ)-net with respect to a uniform metric field g_0 then the point set $\mathcal{P}' = \{F_0 p_i, p_i \in \mathcal{P}\}$, where F_0 is the square root of g_0, is also a δ-power protected (ε, μ)-net, but with respect to the Euclidean metric. We can deduce an upper bound on the sizing field of $\mathcal{C}$ for a uniform metric field using the results of Section 8.3.1 for the Euclidean setting.

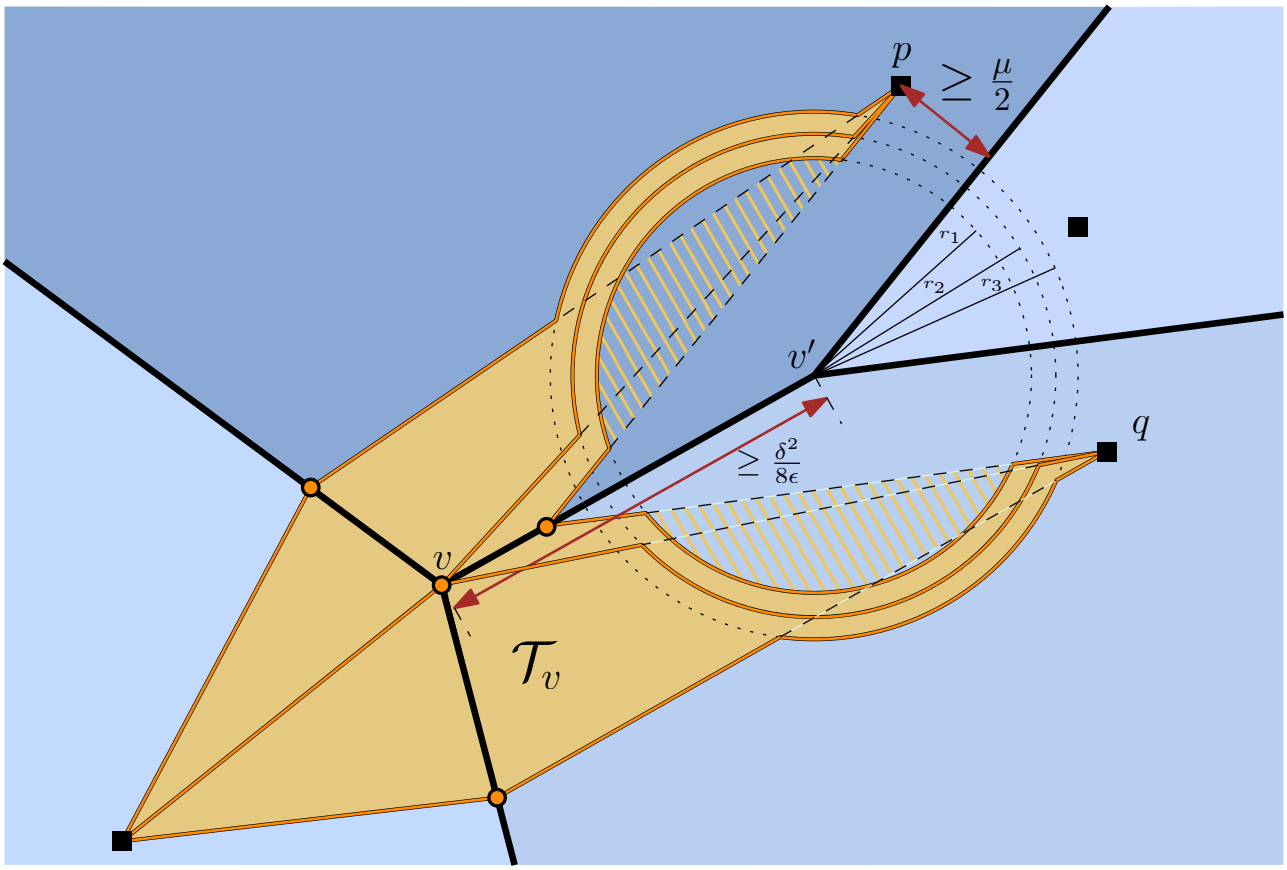

Figure 8.9: Limit case of the deformation of $\mathcal{T}_v$ where the pushing away would create multiple connected components in $\mathcal{T}_v$. Instead, we apply different degrees of "pushing" to preserve the isotopy between $\mathcal{T}_v$ and its deformed version.

Theorem 8.3.10 *Let $\mathcal{P}$ be a finite point set in Ω. Let g be a uniform Riemannian metric field on Ω ($\forall x \in \Omega$, $g(x) = g_0$). Let $\mathcal{C}$ be the canvas, and $h_{\mathcal{C}}$ its sizing field. If*

$$h_{\mathcal{C}} < \left(\min_i \sqrt{\lambda_i}\right) \min\left\{\frac{\mu}{32}, \frac{\delta^2}{128\varepsilon}\right\},$$

then $\mathcal{N}(\mathrm{Vor}_{\mathbb{E}}(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}^d_{\mathbb{E}}(\mathcal{P}))$.

The proof is completely identical to the proof of Theorem 7.3.8 in Chapter III.7 and we do not reproduce it here.

8.3.3 Arbitrary metric field

Finally, we consider an arbitrary metric field g over the domain $\mathbb{R}^n$. The key to proving the equivalence in this setting is to locally approximate the arbitrary metric field with a uniform metric field, a configuration that we have dealt with in the previous section. We shall always compare the metric field g in a neighborhood U with the uniform metric field $g' = g(p_0)$ where $p_0 \in U$. Because g' and the Euclidean metric field differ by a linear transformation, we can simplify matters and assume that g' is the Euclidean metric field. The main argument of the proof will be once again that a power protected net has stable and separated Voronoi vertices.

We now introduce the main result of this section, Theorem 8.3.11.

Theorem 8.3.11 *Let g be an arbitrary metric field on Ω. Assume that $\mathcal{P}$ is a δ-power protected $(\varepsilon, \mu)-$net in Ω with respect to g. Denote by $\mathcal{C}$ the canvas, and $h_{\mathcal{C}}$ its sizing field. If*

$$h_{\mathcal{C}} < \min_{p \in \mathcal{P}} h_{\mathcal{C},p},$$

where $h_{\mathcal{C},p}$ is given by Lemma 8.3.12, and if ε is sufficiently small and δ is sufficiently large (both values will be detailed in the proof), then $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}_g(\mathcal{P}))$.

We prove Theorem 8.3.11 by computing for each point $p \in \mathcal{P}$ the necessary sizing field of the canvas such that the Voronoi cell $\mathrm{V}_g(p)$ is captured correctly. Conditions on ε and δ shall emerge from the intermediary results on the stability of power protected nets.

Lemma 8.3.12 *Let $\psi_0 \geq 1$ be a bound on the metric distortion and g be a Riemannian metric field on U. Let U be an open neighborhood of $\Omega = \mathbb{R}^n$ that is included in a ball $B_g(p_0, r_0)$, with $p_0 \in U$ and $r_0 \in \mathbb{R}^+$ such that $\forall p \in B(p_0, r_0), \psi(g(p), g_{\mathbb{E}}(p)) \leq \psi_0$. Let $\mathcal{P}_U$ be a finite point set in U and let $p_0 \in \mathcal{P}_U$.*

Suppose that $\mathcal{P}_U$ is a δ-power protected $(\varepsilon, \mu)-$net of with respect to g. Let $\mathrm{V}_g(p_0)$ be the Voronoi cell of p_0 in $\mathrm{Vor}_g(\mathcal{P}_U)$. If

$$h_{\mathcal{C},p_0} < \min_i \left(\sqrt{\lambda_i}\right) \min\left\{\frac{\mu}{6}, \frac{\ell_0}{4}\right\},$$

with $\{\lambda_i\}$ the eigenvalues of g_0 and ℓ_0 made explicit in the proof, and if ε is sufficiently small and δ is sufficiently large (both values will be detailed in the proof), then $\mathcal{N}(\mathrm{Vor}_g^d(\mathcal{P})_U) \cong \mathcal{N}(\mathrm{Vor}_g(\mathcal{P})_U)$.

Approach

For readability, the many intermediary results needed to prove Theorem 8.3.11 are moved to Chapter 3. We refer to them at appropriate times.

Our approach is identical to the one used to prove the equivalent result in the two-dimensional setting, but the intermediary steps demand a lot more work. As in Chapter III.7, we use the fact that a Riemannian Voronoi cell $\mathrm{V}_g(p_0)$ can be encompassed into two Euclidean Voronoi cells $\mathrm{DV}_{\mathbb{E}}^{+\eta}(p_0)$ and $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$ that are scaled up and down versions of $\mathrm{V}_{\mathbb{E}}(p_0)$ (Lemma 3.4.12 in Chapter 3). Specifically, $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$ and $\mathrm{DV}_{\mathbb{E}}^{+\eta}(p_0)$ are defined by

$$\mathrm{EV}_{\mathbb{E}}^{-\omega}(p_0) = \{x \in \mathrm{V}_{\mathbb{E}}(p_0) \mid d_{\mathbb{E}}(x, \partial\, \mathrm{V}_{\mathbb{E}}(p_0)) > \omega\},$$

and

$$\mathrm{DV}_{\mathbb{E}}^{+\omega}(p_0) = \bigcap_{i \neq 0} H^{\omega}(p_0, p_i),$$

where $H^{\omega}(p_0, p_i)$ is the half-space containing p_0 and delimited by the bisector $\mathrm{BS}(p_0, p_i)$ translated away from p_0 by ω_0. The constant η is the thickness of this encompassing and depends on the bound on the distortion ψ_0 in the neighborhood, and on the sampling and separation parameters ε and μ. We have that η goes to 0 as ψ_0 goes to 1.

The (local) stability of the power protected nets assumption is proved here again (Lemmas 3.4.2 and 3.4.16 in Chapter 3). From this observation, we can deduce that the Voronoi vertices of the Euclidean Voronoi cell $\mathrm{V}_{\mathbb{E}}(p_0)$ are separated, and thus that the Voronoi vertices of $\mathrm{EV}_{\mathbb{E}}(p_0)$ are separated. A bound on the sizing field can then be computed such that $\mathrm{EV}_{\mathbb{E}}(p_0)$ is captured and thus $\mathrm{V}_g(p_0)$ is captured.

Difficulties The increased difficulty almost entirely comes from proving the stability of the assumption of power protection under metric perturbation in any dimension, that is proving that if we assume that $\mathcal{P}$ is δ-power protected with respect to the arbitrary metric field g, then, in a small neighborhood around p_0, the point set $\mathcal{P}$ is δ_0-power protected with respect to $g_0 = g(p)$. In the two-dimensional case, the stability is obtained by proving that the Voronoi vertices are stable (Lemma 3.4.13), which is itself proved by using separation bounds between Voronoi vertices (Lemma 3.2.4) and bounds on the smallest angles of simplices (Lemma 3.3.1). However, while in 2D we can obtain bounds on such angles simply through the assumption that the point set is a net, this does not extend to higher dimensions due to slivers: we cannot bound the dihedral angles of simplices built from a net. Nevertheless, assuming a power protected net does give us some bounds, but creates a tricky circular dependency as the coefficient δ appears in the dihedral angles (Lemma 3.3.2). We remedy this issue by proving that Euclidean dihedral angles are bounded assuming power protection with respect to the arbitrary metric field, with Lemmas 3.3.6 and 3.3.7.

Proof of Lemma 8.3.12

Lemma 3.4.5 in Chapter 3 gives us that $\mathrm{V}_g(p_0)$ lies in $\mathrm{DV}_{\mathbb{E}}^{+\eta}(p_0)$ and contains $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$. Since $\mathrm{V}_g(p_0)$ contains $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$, if $h_{\mathcal{C}}$ is small enough such that $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$ is captured, then $\mathrm{V}_g(p_0)$ is also captured. Proving that $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$ is captured is done similarly to the Euclidean setting. While we do not explicitly have the power protected net property for the relaxed Voronoi cells (and specifically, $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$), we can still extract the critical property that the Voronoi vertices are separated, as shown by the next lemma.

Lemma 8.3.13 *Assume U, g, and ψ_0 as in Lemma 8.3.12. Assume that the point set $\mathcal{P}_U$ is a δ_0-power protected (ε, μ)-net with respect to the Riemannian metric field g. Then the Voronoi vertices of $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$ are separated.*

Proof. By Lemmas 3.4.2 and 3.4.16 in Chapter 3, we have local stability of the power protection and net properties. Hence, $\mathcal{P}$ is δ_0-power protected (ε_0, μ_0)-net with respect to $g_{\mathbb{E}}$ in U.

Let $L_0 = \frac{\delta_0^2}{4\varepsilon_0}$ be the separation bound induced by the δ_0-power protection property of $\mathcal{P}_U$ (see Lemma 3.2.4 in Chapter 3). Let l be the distance between any two adjacent Voronoi vertices of $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$. We know by Lemma 3.4.14 that the parallelotopic region around a Voronoi vertex is included in a ball centered on the Voronoi vertex and of radius χ. The protection parameter ι is given by $\delta = \iota\varepsilon$. We have that

$$\begin{aligned} l &\geq L - 2\chi \\ &\geq \frac{\delta_0^2}{4\varepsilon_0} - 2\frac{\chi_2}{\sin^{n-2}\left(\frac{\varphi}{2}\right)} \\ &\geq \frac{\delta_0^2}{4\varepsilon_0} - 2\frac{4\eta}{\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0} - \sqrt{1-s_0}\right)^{n-2}} =: \ell_0, \end{aligned} \tag{8.1}$$

where φ represents the dihedral angle and $s_0 = \frac{1}{2}\left[\frac{\iota^2}{4\psi_0^2} - \frac{1}{2}\left(\psi_0^2 - \frac{1}{\psi_0^2}\right)\right]$. For the stability regions not to intersect, we require l to be positive. This can be ensured by enforcing that the lower bound is positive:

$$\frac{\delta_0^2}{4\varepsilon_0} > \frac{8\eta}{\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}} - \sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0} - \sqrt{1-s_0}\right)^{n-2}}$$

Recall that $\varepsilon_0 = \psi_0\varepsilon$, $\mu_0 = \lambda\varepsilon/\psi_0$ and $\rho_0 = 2\psi_0\varepsilon$. Using these notations, we see that $l > 0$ if

$$\begin{aligned}
\delta_0^2 &> \frac{32\varepsilon_0\rho_0^2(\psi_0^2-1)}{\mu_0\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}}-\sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0}-\sqrt{1-s_0}\right)^{n-2}} \\
\delta_0^2 &> \frac{128\varepsilon^2\psi_0^4(\psi_0^2-1)}{\lambda\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}}-\sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0}-\sqrt{1-s_0}\right)^{n-2}}.
\end{aligned} \tag{8.2}$$

This condition is easy to satisfy when ψ_0 goes to 1 because the right hand side of Inequality (8.2) is proportional to $(\psi_0^2-1)\varepsilon^2$.

The intermediary results that we use already impose some conditions on δ and ε, and we thus would like to give the condition in Equation 8.2 in terms of δ, so that it may be compared with Inequality (3.8) (as was done in Remark 7.3.13 in Chapter III.7). In Lemma 3.4.16, we have seen that

$$\delta_0^2 = \frac{\delta^2}{\psi_0^2} + \left(\frac{1}{\psi_0^2}-\psi_0^2\right)(\varepsilon+\chi)^2 - \frac{4\varepsilon\chi}{\psi_0^2},$$

Thus

$$\frac{\delta^2}{\psi_0^2} + \left(\frac{1}{\psi_0^2}-\psi_0^2\right)(\varepsilon+\chi)^2 - \frac{4\varepsilon\chi}{\psi_0^2} > \frac{128\varepsilon^2\psi_0^4(\psi_0^2-1)}{\lambda\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}}-\sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0}-\sqrt{1-s_0}\right)^{n-2}},$$

which is equivalent to

$$\begin{aligned}
\delta^2 &> \frac{128\varepsilon^2\psi_0^6(\psi_0^2-1)}{\lambda\left(\sqrt{1+\frac{\lambda}{2\psi_0^2}}-\sqrt{1-\frac{\lambda}{2\psi_0^2}}\right)\left(\sqrt{1+s_0}-\sqrt{1-s_0}\right)^{n-2}} \\
&\quad + 4\varepsilon\chi \\
&\quad + (\psi_0^4-1)(\varepsilon+\chi)^2. \\
\iff \delta^2 &> 8\varepsilon\psi_0^3\chi + 4\varepsilon\chi + (\psi_0^4-1)(\varepsilon+\chi)^2.
\end{aligned} \tag{8.3}$$

This bound is again proportional to $(\psi_0^2-1)\varepsilon$ and is very similar to the bound given by Inequality (3.8), made explicit in Inequality (3.9), but Inequality (8.3) provides the tougher bound due to the $(\varepsilon+\chi)$ coefficient (see Remark 7.3.13 in Chapter III.7).

□

We can now provide a bound on the sizing field so that it captures $\mathrm{EV}_{\mathbb{E}}^{-\eta}(p_0)$ and prove Lemma 7.3.10. From Section 8.3.1, we have that a sizing field $h_{\mathcal{C}} = \min\{\frac{\mu_0}{32}, \frac{\delta_0^2}{128\varepsilon_0}\}$ ensures the capture of $\mathrm{V}_{\mathbb{E}}(p_0)$ as $\mathcal{P}$ is a δ_0-power protected (ε_0, μ_0)-net with respect to the Euclidean metric field. As we want to capture $\mathrm{EV}_{\mathbb{E}}(p_0)$, we cannot directly use this result. We have nevertheless obtained the separation between the Voronoi vertices of the eroded Voronoi cell (Equation 8.1). It is then straightforward to modify the result of Theorem 8.3.8 by using the separation bound provided in Lemma 8.3.13 instead of the one provided by Lemma 3.2.4. We thus choose

$$h^0_{\mathcal{C},p_0} = \min\left\{\frac{\mu}{6}, \frac{\ell_0}{4}\right\}.$$

Remark 8.3.14 *We here ignore the consequences of Remark 8.3.1 as they just complicate formulas without changing the logical steps.*

We should not forget that we have assumed that g_0 is the Euclidean metric field, which is generally not the case, we must in fact proceed like for the case of a uniform metric field (Theorem 7.3.8):

$$h_{\mathcal{C},p_0} < \min_j \left(\sqrt{\lambda_i}\right) (h^0_{\mathcal{C},p_0}),$$

with $\{\lambda_i\}$ the eigenvalues of g_0.

Therefore, if the seed set satisfies the previous conditions on ε and δ and the canvas honors the sizing field $h_{\mathcal{C},i}$, then $\mathrm{EV}^{-\eta}_{\mathbb{E}}(p_0)$ is captured, and thus $\mathrm{V}_g(p_0)$ is captured, which proves Lemma 7.3.10.

Taking the minimum of all the sizing field values over all $p_i \in \mathcal{P}$, we obtain an upper bound on the sizing field,

$$h_{\mathcal{C}} = \min_{p_i \in \mathcal{P}} h_{\mathcal{C},p_i},$$

such that all the Voronoi cells are captured.

In all the Lemmas necessary to obtain the local results, we have imposed conditions on ε and δ. Similarly to $h_{\mathcal{C}}$, the domain-wide bounds on ε and δ are computed from the local values of ε and δ.

Finally, this proves that $\mathcal{N}(\mathrm{Vor}^d_g(\mathcal{P})) \cong \mathcal{N}(\mathrm{Vor}_g(\mathcal{P})_U)$ in the general setting (Theorem 8.3.1), when geodesics are exactly computed.

8.3.4 Extension to surfaces

The previous results may be generalized to n-manifolds embedded in $\mathbb{R}^m$. We shall assume that, apart from the metric induced by the embedding of the domain in Euclidean space, there is a second metric g defined on $\mathcal{M}$. Let $\pi_p : \mathcal{M} \to T_p\mathcal{M}$ be the orthogonal projection of points of $\mathcal{M}$ on the tangent space $T_p\mathcal{M}$ at p. For a sufficiently small neighborhood $U_p \subset T_p\mathcal{M}$, π_p is a local diffeomorphism (see Niyogi [111]).

The pullback of the metric g with the inverse projection π_p^{-1} defines a metric on U_p. We have a metric on a subset of a n-dimensional space, in this case the tangent space, giving us a setting that we have already solved. Boissonnat et al. [17, Lemma 3.7] give bounds on the metric distortion of the projection on the tangent space. This result allows to translate the δ-power protected (ε, μ)-net properties between the tangent space and the manifold. Details can be found in Section 7.3.4 in Chapter III.7, where the reasoning is similar.

8.3.5 Approximate geodesic distance computations

We have so far assumed that geodesics are computed exactly, but this is not the case in practice. Nevertheless, once the error in the approximation of the geodesic distances is small enough, the computation of the discrete Riemannian Voronoi diagram with approximate geodesic distances can be equivalently seen as the computation of a second discrete Riemannian Voronoi diagram with a slightly different metric field and using exact geodesic distances. This is achieved exactly as in Section 7.3.5 in Chapter III.7 and we thus do not reproduce it here.

8.4 Curved Riemannian Delaunay triangulation

For each simplex σ in the Delaunay complex, we can consider its curved realization (see Section 2.9 in Chapter 2). The definition of the curved realization is based on the Riemannian center of mass [83, 65], also known as a Karcher mean [85]. This step of our theory fully relies on the work of Dyer et al. [65], and we only summarize here their results.

Definition 8.4.1 *The curved realization of a Delaunay simplex, $\widetilde{\sigma}$, is said to be non-degenerate if, and only if, it is homeomorphic to $\boldsymbol{\phi^n}$.*

The following theorem gives conditions such that simplices are non-degenerate.

Theorem 8.4.2 (Non-degeneracy criteria) *Suppose that $\mathcal{M}$ is a Riemannian manifold with sectional curvatures K bounded by $|K| \leq \Lambda$, and $\sigma_{\mathcal{M}}$ is a curved Riemannian simplex, with $\sigma_{\mathcal{M}} \subset B_\rho \subset \mathcal{M}$, where B_ρ is an open geodesic ball of radius ρ, with*

$$\rho \leq \rho_0 = \min\left\{\frac{\iota_{\mathcal{M}}}{2}, \frac{\pi}{4\sqrt{\Lambda}}\right\}.$$

Then $\sigma_{\mathcal{M}}$ is non-degenerate if there is a point $p \in B_\rho$ such that the lifted Euclidean simplex $\sigma(p)$ has thickness satisfying

$$t(\sigma(p)) > 10\sqrt{\Lambda}L(\sigma_{\mathcal{M}}),$$

where $L(\sigma_{\mathcal{M}})$ is the geodesic length of the longest edge in $\sigma_{\mathcal{M}}$.

This can then be extended to a domain-wide result with the following definition and theorem.

Definition 8.4.3 *If p is a vertex in an abstract simplicial complex $\mathcal{A}$, we define the star of p to be the subcomplex $\mathrm{S}(p)$ of $\mathcal{A}$ consisting of all simplices that contain p, together with the faces of these simplices. The underlying topological space (or carrier) of a complex $\mathcal{A}$ is denoted $|\mathcal{A}|$. We say that $\mathrm{S}(p)$ is a full star if $|\mathrm{S}\,p|$ is a closed topological ball of dimension n with p in its interior, and $\mathcal{A}$ contains no simplices of dimension greater than n.*

Theorem 8.4.4 (Triangulation criteria) *Suppose $\mathcal{M}$ is a compact n-dimensional Riemannian manifold with sectional curvatures K bounded by $|K| \leq \Lambda$, and $\mathcal{A}$ is an abstract simplicial complex with finite vertex set $\mathcal{S} \subset \mathcal{M}$. Define a quality parameter $t_0 > 0$, and let*

$$h = \min\left\{\frac{\iota_{\mathcal{M}}}{2}, \frac{\sqrt{n}t_0}{6\sqrt{\Lambda}}\right\}.$$

If,

- *for every $p \in \mathcal{S}$, the vertices of $St(p)$ are contained in $B_{\mathcal{M}}(p;h)$, and the balls $\{B_{\mathcal{M}}(p;h)\}_{p\mathcal{S}}$ cover $\mathcal{M}$,*
- *for every $p \in \mathcal{S}$, the restriction of the inverse of the exponential map $\exp_p^{-1}$ to the vertices of $St(p) \subset \mathcal{A}$ defines a piecewise linear embedding of $|St(p)|$ into $T_p\mathcal{M}$, realizing $St(p)$ as a full star such that every simplex $\sigma(p)$ has thickness $t(\sigma(p)) \geq t_0$,*

then $\mathcal{A}$ triangulates $\mathcal{M}$, and the triangulation is given by the barycentric coordinate map on each simplex.

8.5 Straight realization of the Riemannian Voronoi diagram

An anisotropic simplicial complex with straight simplices is usually desired for practical applications. Starting from the curved Riemannian Delaunay triangulation, we give conditions such that "straightening" the geodesic edges of the curved Riemannian Delaunay triangulation preserves the property of being an embedded dual.

We first prove that, under some sampling conditions, the facet of a curved Riemannian simplex and its straight equivalent are close from one another. This is a generalization of a result that we have established in the planar and surface settings in Chapter III.7. The proof of Lemma 8.5.1 employs however a different approach. The proximity between a curved Riemannian simplex and its straight equivalent allows to then prove that "straightening" simplices creates no inversion of simplices or self-intersections. This is achieved in Theorem 8.5.2.

8.5.1 Proximity between straight and curved Riemannian simplices

We now prove that two points p and q with the same barycentric coordinates on respectively the geodesic facet and the straight facet are close.

Lemma 8.5.1 *Let $\{p_i\}$ be a set of $n+1$ vertices in Ω such that $D = \cap_{p_i \in \mathcal{P}} V(p_i) \neq \emptyset$. Let $\bar{\sigma}$ and $\widetilde{\sigma}$ be the straight and curved Riemannian simplices that realize D. Let x_g be a point on the curved Riemannian simplex σ_g determined by the barycentric coordinates $\{\lambda_i\}$ (see Section 2.9). Let x_e be the point uniquely determined by $\{\lambda_i\}$ as $x_e = \sum_i \lambda_i p_i$. If the geodesic distance d_g is close to $d_{\mathbb{E}}$, then*

$$|x_g - x_e| \leq \sqrt{2 \cdot 4^3 (\psi_0 - 1)\varepsilon^2}.$$

Proof. The key observation is that given a convex function f and a function f' that is close, that is $f - f' < \alpha$ with α small, then the minimum value of f' is at most of $\min f + \alpha$. If we observe that at any point x where $f(x) > \min f + 2\alpha$, we also have $f'(x) > \min f + \alpha$ so x is not a minimum of f', we see that the minima of f and f' can not be far apart. In particular, we have that if $x_{f',\min}$ is the point where f' attains its minimum, then $f'(x_{f',\min}) \leq \min f + \alpha$. The precise argument goes as follows.

We again assume that (possibly after a linear transformation) the metric is close to the Euclidean one, that is:

$$d_g(x, y) = |x - y| + \delta d_g(x, y),$$

with $|\delta d_g(x, y)| \leq (\psi_0 - 1)\,|x - y|$. If we assume that $|x - y| \leq 4\varepsilon$ and $\psi_0 \leq 2$, it follows that

$$d_g(x, y)^2 = |x - y|^2 + \delta d_g^2(x, y),$$

with

$$\delta d_g^2(x, y) \leq 4^3 (\psi_0 - 1)\varepsilon^2.$$

Recall that x_g is the point where the functional

$$\mathcal{E}_\lambda(x) = \sum_i \lambda_i d_g(x, p_i)^2$$

attains its minimum.

Using the bounds above, we find that

$$\sum_i \lambda_i d_g(x, p_i)^2 = \sum_i \lambda_i \,|x - p_i|^2 + \sum_i \lambda_i \delta d_g^2(x, p_i),$$

where $\sum_i \lambda_i \delta d_g^2(x, p_i) \leq 4^3 (\psi_0 - 1)\varepsilon^2$. We also see that

$$\left| \sum_i \lambda_i d_g(x_g, p_i)^2 - \sum_i \lambda_i \,|x_g - p_i|^2 \right| \leq 4^3 (\psi_0 - 1)\varepsilon^2.$$

Taking f' to be $\sum_i \lambda_i d_g(x_g, p_i)^2$ in the explanation above, we find that

$$\left| \sum_i \lambda_i d_g(x_g, p_i)^2 - \sum_i \lambda_i \left| \sum_j \lambda_j v_j - p_i \right|^2 \right| \leq 4^3(\psi_0 - 1)\varepsilon^2,$$

because the Euclidean barycenter $x_e = \sum_i \lambda_i p_i$ is where the function $\sum_i \lambda_i |x_g - p_i|^2$ attains its minimum. Combining these results yields

$$\left| \sum_i \lambda_i \left| x_g - p_i \right|^2 - \sum_i \lambda_i \left| \sum_j \lambda_j v_j - p_i \right|^2 \right| \leq 2 \cdot 4^3(\psi_0 - 1)\varepsilon^2. \tag{8.4}$$

This bounds the distance between x_g and x_e. An explicit bound can be found by observing that

$$\begin{aligned}
\sum_i \lambda_i \left| x - p_i \right|^2 &= \sum_i \lambda_i \left(\left(x^1 - p_i^1 \right)^2 + \ldots + (x^n - p_i^n)^2 \right) \\
&= \sum_i \lambda_i \left(x^1 - p_i^1 \right)^2 + \sum_i \lambda_i \left(x^2 - p_i^2 \right)^2 + \ldots + \sum_i \lambda_i \left(x^n - p_i^n \right)^2 \\
&= \left(x^1 - \sum_i \lambda_i p_i^1 \right)^2 - \left(\sum_i \lambda_i p_i^1 \right)^2 + \sum_i \lambda_i \left(p_i^1 \right)^2 + \ldots \\
&\qquad + \left(x^n - \sum_i \lambda_i p_i^n \right)^2 - \left(\sum_i \lambda_i p_i^n \right)^2 + \sum_i \lambda_i \left(p_i^n \right)^2 \\
&= \left| x - \sum_i \lambda_i p_i \right|^2 + \sum_j \left[- \left(\sum_i \lambda_i p_i^j \right)^2 + \sum_i \lambda_i \left(p_i^j \right)^2 \right].
\end{aligned} \tag{8.5}$$

Then, applying Equation 8.5 for both both $x = x_g$ and $x = x_e = \sum_j \lambda_j p_j$ in Equation (8.4), we obtain:

$$\begin{aligned}
&\left| \left| x_g - \sum_i \lambda_i p_i \right|^2 + \sum_j \left[- \left(\sum_i \lambda_i p_i^j \right)^2 + \sum_i \lambda_i \left(p_i^j \right)^2 \right] \right. \\
&\quad \left. - \left(\left| \sum_j \lambda_j p_j - \sum_i \lambda_i p_i \right|^2 + \sum_j \left[- \left(\sum_i \lambda_i p_i^j \right)^2 + \sum_i \lambda_i \left(p_i^j \right)^2 \right] \right) \right| \\
&\qquad = \left| x_g - \sum_i \lambda_i p_i \right|^2 \leq 2 \cdot 4^3(\psi_0 - 1)\varepsilon^2.
\end{aligned}$$

which yields a distance bound of $\sqrt{2 \cdot 4^3(\psi_0 - 1)\varepsilon^2}$. □

As expected, the bound goes to 0 when the distortion goes to 1 or when the sampling gets dense.

Although we have formulated this metric distortion result for simplices, the same proof extends almost verbatim to continuous distributions. By this we mean that the barycenter with respect to a metric g of a continuous distribution is close to the barycenter with respect to the Euclidean metric, if g is close to the Euclidean metric. Furthermore, note that the proof does not depend on the weights being positive.

8.5.2 Embeddability of the straight Riemannian Delaunay triangulation

We apply Lemma 8.5.1 to the facets of the simplices of $\widetilde{\mathrm{Del}}_g(\mathcal{P})$. Recall that the altitude of p_i in σ is noted $D(p_i, \sigma)$.

Theorem 8.5.2 *Let $\{p_i\}$ be a set of $n+1$ vertices in Ω that define $D = \cap_{p_i \in \mathcal{P}} \mathrm{V}(p_i) \neq \emptyset$. Let σ and σ_g be the straight and curved Riemannian simplices that realize D. Let τ_g be a facet of σ_g, opposite of the vertex p_i. If for all $x_g \in \tau_g$, $|x_g - x_e| \leq D(p_i, \sigma)$, then there is no inversion created when σ_g is straightened onto σ.*

If this condition is fulfilled for all $\sigma_g \in \widetilde{\mathrm{Del}}_g(\mathcal{P})$, then $\overline{\mathrm{Del}}_g(\mathcal{P})$ is embedded.

Proof. The lower bound on $D(p, \sigma)$ given in Lemma 3.3.2 in Chapter 3 is proportional to ε. The proximity (upper) bound of Lemma 8.5.1 is proportional to $\sqrt{(\psi_0 - 1)}\varepsilon$, therefore going to 0 much faster. The embeddability is thus satisfied once

$$\sqrt{2 \cdot 4^3 (\psi_0 - 1)\varepsilon^2} < A_\iota \nu \iota^2 \varepsilon$$
$$\psi_0 < 1 + \frac{\left(A_\iota \nu \iota^2\right)^2}{2 \cdot 4^3},$$

where A_ι and ν are constants whose definitions can be found in Lemma 3.3.2 in Chapter 3. □

8.6 Construction of power protected nets

Our theory introduces requirements on the canvas $\mathcal{C}$ (through its sizing field) and on the point set $\mathcal{P}$ such that the two duals of the discrete Riemannian Voronoi diagrams are triangulations. The canvas must be sufficiently dense, and the seed set a δ-power protected (ε, μ)-net with specific values of δ and ε. The former requirement is easily achieved through refinement. A refinement algorithm to construct power protected nets was described in Section 5.4 in Chapter II.5.

In the two-dimensional setting, neither the sampling nor the canvas needed to be as dense in practice as the theory required them to be. Instead, a farthest point refinement was used to generate the point set. Results in Chapter III.7 have shown that this is not the case in higher dimensions. The lack of power protection was shown to be the culprit of inverted elements and power protection is thus necessary. Unfortunately, the refinement algorithm proposed in Section 5.4 in Chapter II.5 is mostly theoretical and the runtime of such algorithm would be unreasonable.

8.7 Conclusion

We have proved in this chapter that the discrete Riemannian Voronoi diagram and the related algorithms introduced in Chapter III.6 are theoretically sound in any dimension. These guarantees come in the shape of a quality requirement on the point set, through the power protected net assumption, and on the density on the canvas. Although we followed the same approach to prove the theoretical soundness as the previous chapter, many new intermediary results were required.

As in the two-dimensional setting, it would be desirable to expose quality bounds on the simplices and to estimate the loss of quality that results from straightening curved Riemannian simplices into straight Riemannian simplices. Finally, devising a practical algorithm to generate power protected nets ought to be an interesting subject of research, with various areas of applications.

Acknowledgment

The use of Sperner's lemma was suggested by Ramsay Dyer.

9. Conclusion and perspectives

This thesis has studied, introduced and investigated a number of anisotropic mesh generation algorithms, all based on the well-known structures of the Delaunay triangulation and the Voronoi diagram. We have sought approaches that combine theoretical soundness, that is the capacity to prove that the desired result is obtained under certain conditions, and practicality. Our results are mixed: while we have proposed a number of methods that were all provably correct, none of these methods were universally practical.

The practical side of locally uniform anisotropic meshes, a framework of theoretical mesh generation that precedes this thesis, has been thoroughly investigated. We have proposed a generic, robust, and fast implementation based on the CGAL library. Although we obtain anisotropic meshes that conform to the metric field, the lack of control on the resolution of inconsistencies generally created unreasonably large – and thus impractical – meshes. This shortcoming was attributed to the inherent rigidity in the construction of the stars as only one metric is considered. We attempted to weaken this rigidity and improve the results by modifying and combining another star-based approach, the tangential Delaunay complex, with an alternative construction of the anisotropic Voronoi diagram of Labelle and Shewchuk. Unfortunately, the results were similar to those obtained in the study locally uniform anisotropic meshes. Finally, increasing again the quantity of metric information considered in our algorithms, we have looked to Riemannian Voronoi diagrams, an expensive but the most accurate diagram with respect to the metric field. We introduced a discrete structure on top of which we compute geodesics to accurately capture the combinatorial information of the Riemannian Voronoi diagram. From a quality point of view, we obtained much better results for two-dimensional manifolds than we have with our previous approaches. We are able to build visually pleasing meshes that are closely conforming to the metric field, especially in difficult regions where other mesh techniques would fail or create isotropic elements. Additionally, the use of a Riemannian Voronoi diagram allowed us to build curved Riemannian Delaunay triangulations. Although Riemannian Voronoi diagrams require fewer points to have an embedded dual than other anisotropic Voronoi-based methods, this advantage only lasts for a time before every method yields similar results. Despite the use of anisotropic star sets generated from locally uniform anisotropic meshes, the running time of our diagram is unreasonably slow for large meshes and three and higher-dimensional domains. Finally, our farthest point refinement algorithm does not generate seed set whose dual is a triangulation in the setting of three-dimensional domains, and the use of power protected seed sets seems necessary.

The quintessence of the proofs of our algorithms has been to ensure that adjacent Voronoi vertices are far from one another by assuming that we use power protected nets. We also proposed a refinement algorithm to generate such point sets in any dimension. The strength and versatility of power protected nets allowed us to give conditions such that the anisotropic Voronoi diagram of Labelle and Shewchuk (and by extension multiplicative Voronoi diagrams) have an embedded dual. Proving the theoretical soundness of our algorithms required a large number of interesting intermediary results, for example on the stability of the hypothesis of power protection and net under metric perturbations and on the quality of the Delaunay and Voronoi structures created from power protected nets.

Future work

Different paths can be followed to extend this work.

Firstly, all of our theoretical work has relied on power protected nets and the generation of such point sets is in practice a difficult task as it is generally complicated to estimate problematic configurations beforehand. In our implementation of the framework of locally uniform anisotropic meshes, we have used a trial and error approach to construct such point sets, but although this method did (eventually) construct good point sets, the running time was a big factor in the impracticality of the algorithm. Devising a practical way to construct power protected nets would provide a major improvement to all of our methods. Additionally, we are optimistic that their use could generate good anisotropic triangulations as dual of discrete Riemannian Voronoi diagrams and even as duals of anisotropic Voronoi diagram using the distance of Labelle and Shewchuk.

Still from a theoretical point of view, various tasks require more work. First of all, our proofs do not at the moment handle manifolds with boundaries, which an arbitrary "real" domain contains. A comprehensive study of the quality of the elements produced by our algorithms is also lacking. As most of our methods are based on Delaunay refinement and the insertion of points near the circumcenter of a simplex, it would be interesting to investigate theoretical bounds on the angle and shape of simplices.

Further away from our work is the question of the feasibility of a metric field: when is a unit mesh physically possible? Our choice of sizing field in the metric fields that we have considered has always been an estimation, and a value based on solider theoretical grounds would be useful.

Tangential complex embedding

The original tangential Delaunay complex has shown to be an effective structure in the context of (isotropic) manifold reconstruction. It can thus be concluded that the impracticality of our tangential complex-based approach lies entirely within our modifications of the original structure to the anisotropic setting. Despite this shortcoming, the tangential Delaunay structure could still be potentially used for the generation of anisotropic meshes by embedding or transforming the input domain to a new domain where an isotropic mesh is required.

The Nash's embedding theorem – which provides an isometric embedding – naturally comes to mind. However, the embedding introduced by Nash is very theoretical: the concept of infinitely small and recursive twists to the manifold do not form a domain that can be easily meshed. Authors have previously considered the idea of embedding the input domain to ease the process of meshing: Zhong et al. [145] used conformal embedding and Panozzo et al. [114] warped input surfaces to produce anisotropic quadrangulations. However, these methods are limited by the dimension of the embedding spaces to keep a reasonable runtime. The use of the tangential complex would lift such limitation as the complexity of the tangential complex almost entirely depends on the intrinsic dimension of the embedded manifold and not the dimension of the embedding space.

Finite element analysis

Finite element method (FEM) is a renowned numerical technique to solve partial differential equations that makes use of meshes. Recent developments have extended the use of these methods to curved meshes, whose faces are often described by Bézier curves and patches. Another line of research could be the adaptation of the finite element method to curved Riemannian elements. Indeed, curved Riemannian simplices are defined through a pointwise minimization process based on a (λ_i) barycentric coordinate vector and we thus immediately have a pointwise deformation between a curved Riemannian simplex and the traditional reference simplex of the finite element. Since curved Riemannian

elements follow by definition the metric field absolutely, they could provide – in theory – a great level of accuracy.

A. Metrics

This appendix details the analytical expressions or methods used to compute the metric fields used in this work.

A.1 Size of a metric

As explained in Section 2.13 of Chapter 2, the metric field can be seen as directly encoding a sizing field in the amplitude of its eigenvalues. We detail below the various metric fields that we consider in this thesis. All these metric fields are computed by detailing the eigenvectors and the eigenvalues of the matrix at a point p of the domain. Scaling the metric at a point is simply scaling the (inverse of the) eigenvalues. For example if e_1 and e_2 are two eigenvalues of a matrix at a point, considering the same matrix with eigenvalues $e_1/10$ and $e_2/10$ multiplies the radius of the unit sphere at this point by 10.

Remark A.1.1 *The scaling must be identical for all the eigenvalues, otherwise the anisotropy is modified.*

A.2 Curvature

We detail in this section how to compute a metric field induced by the curvature for implicit and polyhedral domains.

A.2.1 Implicit surfaces

Let S be an implicit surface defined by $f(x, y, z) = 0$. At each point p of S, we denote by N the normal vector $-\nabla f / \|\nabla f\|$ and by H the Hessian of f. The principal curvatures $c_{\max}$, $c_{\min}$ are the eigenvalues of the matrix

$$C = P \cdot H' \cdot P,$$
$$\text{where } H' = H \|\nabla f\| \text{ and } P = I_3 - N \cdot N^t,$$

see e.g. [79]. By convention, we define $c_{\max}$ to be the eigenvalue with the maximal absolute value. Let $U_{\max}$, $U_{\min}$ and N be the normalized eigenvectors of C, and let $U = [U_{\max}, U_{\min}, N]$. The metric at the point p is defined by $M_p = F_p^t F_p$, where:

$$\begin{aligned} F_p &= U \cdot \Delta \cdot U^t, \\ \text{with } \Delta &= \text{Diag}\{e_{\max}, e_{\min}, e_{\text{n}}\}, \\ e_{\max} &= \sqrt{|c_{\max}|}, \\ e_{\min} &= \sqrt{|c_{\min}|}. \end{aligned}$$

The value of e_{n} does not significantly affect the shape of the facets of the mesh since those facets are almost tangent to the surface. Experimental evidence indicates that e_{n} should be of the same order of magnitude as the global maximum of $\sqrt{c_{\max}}$.

If one of the curvature values vanishes, we replace it with a small strictly positive value ε such that the metric stays positive definite. The value of ε plays an important role in the final mesh and is difficult to choose, as explained in the next section.

The ε criterion

We first illustrate the influence of the value of ε over the final mesh. The "Chair" surface is used (see Section B.1 in Appendix B). This surface is characterized by the presence of parabolic points along a circle (similarly to the top of a torus) on each of its "side" and in a saddle-type of region near the center of the surface. Figure A.1 illustrates the two different types of parabolic points met on that surface. The positions of parabolic points are distinguishable in Figure A.2 (leftmost) due to the over-refinement around these points.

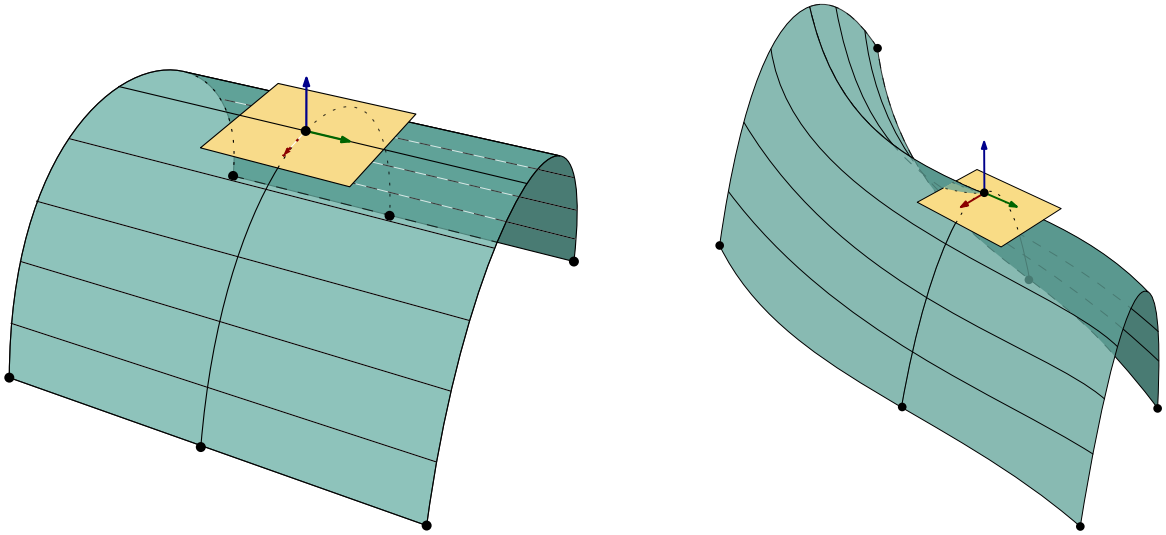

Figure A.1: The two configurations of parabolic points present on the "Chair" surface. In both case, the curvature along the green vector is null.

The different meshes obtained for different values of ε for the "Chair" surface, using the algorithm described in Chapter I.4. The other parameters, and specifically the approximation, are kept constant.

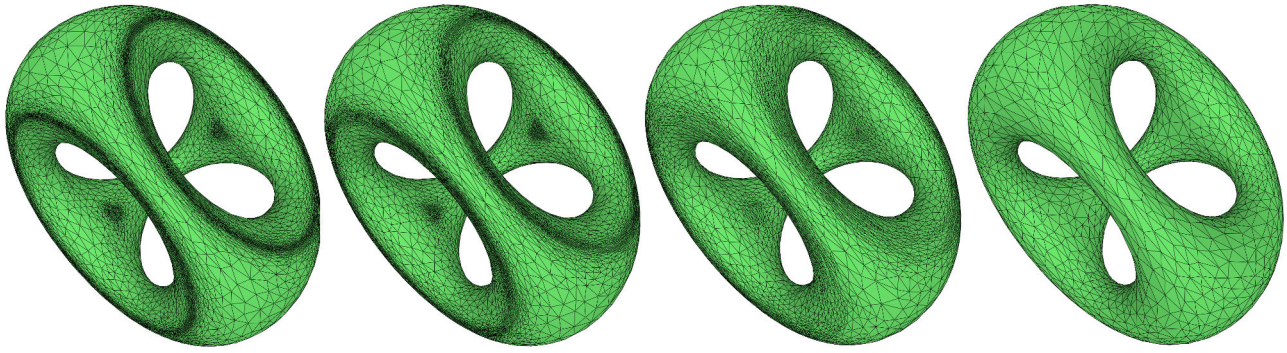

Figure A.2: Meshes produced by the algorithm presented in Chapter I.4 for different values of ε and constant parameters. The value of ε increases from left to right: 0.1, 0.3, 0.5, 0.9.

The rightmost value provides the lowest number of vertices for the set of parameters and is what is generally expected as result. However, the corresponding value of ε was found by trial and error and is completely specific to this surface and to the choice of the other parameters; simply scaling the domain would already change the "optimal" value of ε. Worse, this "optimal" value of ε might not in fact be optimal for both types of parabolic points present on the "Chair" surface. This can be more easily seen on the "cyclide" surface (see Section B.3 in Appendix B). The cyclide surface is a torus with varying small radius, also characterized by the presence of parabolic points along a circle. No single value of ε can be satisfying: if ε is chosen too small, the mesh will be over-refined where the small radius is largest (and possibly over-refined everywhere if ε is really small, like in Figure A.3); if ε is chosen too large, the elements are not as anisotropic as they could be, and there are more elements than an optimal mesh could have.

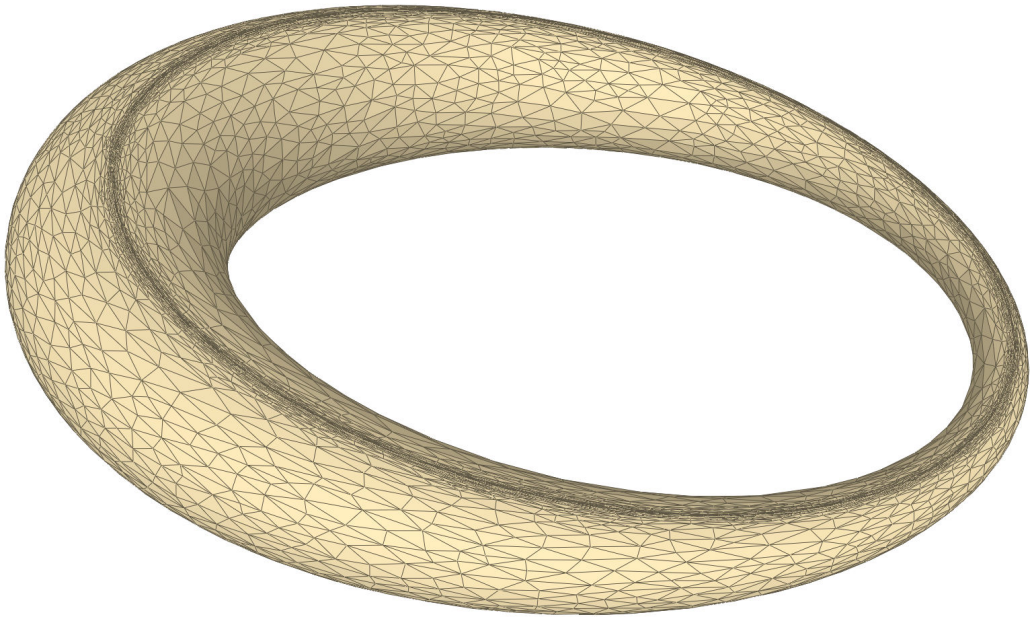

Figure A.3: cyclide surface for a low value of ε.

It would thus be desirable to consider a varying ε, but it is difficult to evaluate what its value should be. We attempted to solve this problem heuristically but no satisfying method was found and this is thus still an open problem.

A.2.2 Polyhedral Surfaces

We can estimate the curvature tensor of $\tilde{S}$ from S. This is done in two steps.

1. **Curvature estimation at the vertices of S**
   We first estimate the curvature tensor at each vertex $p \in S$ as follows. We compute the Euclidean k-nearest vertices of p and then apply the jet-fitting algorithm of [40] to estimate the two principal curvatures $c_{\max}$ and $c_{\min}$, the corresponding unit vectors $U_{\max}$ and $U_{\min}$, and the normal vector N at p.

   From the estimated curvatures, we compute $e_{\max}$ and $e_{\min}$ as in the case of an implicit domain. We choose e_{n} as in the case of implicit domains and store at each vertex p a resulting estimated metric tensor $G(p) = U\Delta^2 U^t$, with $U = [U_{\max}, U_{\min}, N]$ and $\Delta = \mathrm{diag}\{e_{\max}, e_{\min}, e_{\mathrm{n}}\}$.

2. **Blending metric tensors**
   To estimate the metric tensor at any prescribed query point on S, we smoothly blend the metric tensors computed in the previous step as described in Section 2.2.3 in Chapter 2.

A.3 Hyperbolic shock

If E denotes such a scalar field defined in $\mathbb{R}^3$ and Ω a domain, we may want to mesh Ω to provide the best approximation of E on Ω. The triangles are elongated orthogonally to the direction of the gradient ∇E and the density depends both on the norm and the changing rate of the gradient.

We define the metric field as follows :

$$F_M = U \cdot \mathrm{Diag}\{1/(1 + \varphi \, \|\nabla E\|), 1, 1\} \cdot U^t$$

where $U = [\frac{\nabla E}{\|\nabla E\|}, U_1, U_2]$, U_1 and U_2 being two arbitrary unit vectors that form an orthogonal basis of the orthogonal complement of ∇E. If the gradient is zero, the metric field is isotropic. The parameter φ controls how the facets are stretched with respect to the norm of the gradient.

The hyperbolic shock is an arbitrary sine-shaped shock function that can be described by the following equation:

$$E(x, y, z) = \tanh\left(\frac{1}{\lambda}(2x - \sin(5y))\right) + x^3 + xy^2,$$

with $\lambda = 0.6$. The metric field is derived from E as described above.

A.4 Swirl

The Swirl metric field is based on a uniform vector field that creates a swirl to which an anisotropy ratio that depends on the distance to the center is attached. The metric field has a parameter ν that controls the circular motion; we use $\nu = 3$. The metric M at a point $p(x, y)$ is constructed as follows.

First, construct the eigenvectors v_1 and v_2 of the matrix:

$$\begin{aligned}
v_{1,x} &= \frac{-(xs + yc) + (xc - ys)(1 - x^2 - y^2)^2}{r^2} \\
v_{1,y} &= \frac{(xc - ys) + (xs + yc)(1 - x^2 - y^2)^2}{r^2} \\
n &= \sqrt{v_{1,x}^2 + v_{1,y}^2} \\
v_1 &= \left(\frac{v_{1,x}}{n}, \frac{v_{1,y}}{n}\right) \qquad v_2 = \left(\frac{-v_{1,y}}{n}, \frac{v_{1,x}}{n}\right)
\end{aligned}$$

where $r^2 = x^2 + y^2$, $c = \cos(\nu)$ and $s = \sin(\nu)$.

Then construct the eigenvalues e_1 and e_2 (responsible for the anisotropy ratio):

$$\begin{aligned}
f &= \frac{\sqrt[3]{r} + \sqrt{r}}{2} \\
e_1 &= \frac{1}{|50(1 - (f - 1)^2)|}, \qquad e_2 = 1.
\end{aligned}$$

We can now assemble the matrix :

$$\begin{aligned}
U &= (v_1, v_2) \\
\Delta &= \mathrm{Diag}(e_1, e_2) \\
F &= U^t \cdot \Delta \cdot U.
\end{aligned}$$

And finally $M = F^t \cdot F$.

A.5 Starred

The starred metric field is similar to the hyperbolic shock but has more emphasis on long unidirectional anisotropic regions. The metric at $p(x, y)$ is computed as follows:

$$\begin{aligned} v_1 &= 3x^2 - \frac{xy}{5} - y^2 + \frac{3x^2}{10} \\ v_2 &= -\frac{x^2}{10} - 2xy \\ n &= \sqrt{v_1^2 + v_2^2}, \end{aligned}$$

and

$$V = \begin{pmatrix} v_1/n & -v_2/n \\ v_2/n & v_1/n \end{pmatrix}.$$

The anisotropy is controlled by a single variable $a = \max\left\{\alpha, \frac{1}{1+\psi n}\right\}$; we take $\alpha = 0.1$ and $\psi = 1$. The complete metric G_p at p is defined by:

$$G_p = V^t \cdot \begin{pmatrix} a^2 & 0 \\ 0 & 1 \end{pmatrix} \cdot V.$$

A.6 Radial shock

The radial shock metric field can be thought to model the propagations of circular shocks from a pointwise origin outwards. The metric at $p(x, y)$ is computed as follows:

$$\begin{aligned} r &= x^2 + y^2 \\ v_1 &= \frac{x}{\sqrt{r}} \\ v_2 &= \frac{y}{\sqrt{r}}, \end{aligned}$$

and

$$V = \begin{pmatrix} v_1 & -v_2 \\ v_2 & v_1 \end{pmatrix}.$$

Letting $s = \sin(r/5)$ and $c = \cos(r/5)$, the eigenvalues are given by:

$$\begin{aligned} e_1 &= \frac{-5s - 2r(c - 3s^2)}{(rs^2)^{\frac{3}{2}}} \\ e_2 &= \frac{s}{rs^2}. \end{aligned}$$

The complete metric G_p at p is defined by:

$$G_p = V^t \cdot \begin{pmatrix} e_1 & 0 \\ 0 & e_2 \end{pmatrix} \cdot V.$$

A.7 Radial shock (3D)

The 3-dimensional radial shock metric field can be thought to model the propagations of one circular shock from a pointwise origin outwards. The metric at $p(\bar{x}, \bar{y}, \bar{z})$ is computed as follows.

$$\begin{aligned} r &= \sqrt{\bar{x}^2 + \bar{y}^2 + \bar{z}^2} \\ x = \frac{\bar{x}}{r} \quad y &= \frac{\bar{y}}{r} \quad z = \frac{\bar{z}}{r} \\ v_{2y} &= -\frac{yz}{\sqrt{r^2(x^2+y^2)}} \\ v_{2x} &= \frac{v_{2y}x}{y} \\ v_{2z} &= \sqrt{1 - v_{2y}^2\left(1 + \frac{x^2}{y^2}\right)}. \end{aligned}$$

The anisotropy is determined by the following

$$\begin{aligned} &\alpha = 0.025 \\ &e_1 = \alpha + (1 - \exp\left(-0.01\left|r^2 - 0.15\right|\right)) \\ &e_2 = 1 \\ &e_3 = 1. \end{aligned}$$

The maximum anisotropy ratio, at the center of the shock, is given by $1/\alpha$ (thus 40 with the default value of 0.025). The matrix of eigenvectors is given by

$$V = \begin{pmatrix} x & v_{2x} & yv_{2z} - zv_{2y} \\ y & v_{2y} & zv_{2x} - xv_{2z} \\ z & v_{2z} & xv_{2y} - yv_{2x} \end{pmatrix}.$$

Finally, the complete metric G_p at p is defined by:

$$G_p = V^t \cdot \begin{pmatrix} e_1^2 & 0 & 0 \\ 0 & e_2^2 & 0 \\ 0 & 0 & e_3^2 \end{pmatrix} \cdot V.$$

A.8 Unidimensional shock

We denote unidimensional shocks any metric field for which the eigenvectors of the metric are aligned with the axis at all point. In other words, the metrics are diagonal matrices.

Different kind of shocks are possible, but we keep a preference for exponential-based shocks, as they can be easily linked with the Log-Euclidean interpolation. A typical unidimensional shock in 3D might be given by the following eigenvalues at a point $p = (x, y, z)$ of the domain:

$$\begin{aligned} e_1(p) &= 0.1 + (1 - \exp(-|x - 0.6|)) \\ e_2(p) &= 0.3 + (1 - \exp(-|y - 0.7|)) \\ e_3(p) &= 0.05 + (1 - \exp(-|z - 0.2|)). \end{aligned}$$

Meaning, for example, that the anisotropy along (Oy) is maximal at $y = 0.7$, with a stretching of $1/0.3 \approx 3$ and decreases as $|y - 0.7|$ increases.

The complete metric G at p is given by the matrix

$$G = \begin{pmatrix} e_1^{-2} & 0 & 0 \\ 0 & e_2^{-2} & 0 \\ 0 & 0 & e_3^{-2} \end{pmatrix}.$$

B. Domains

B.1 Chair

"Chair" is a non-convex implicit surface defined by the equation:

$$f(x,y,z) = \left(x^2+y^2+z^2-ak^2\right)^2 - b\left((z-k)^2-2x^2\right)\left((z+k)^2-2y^2\right).$$

We use $a = 0.8$, $b = 0.4$, $k = 1.0$.

B.2 Cow

"Cow" is a rough polyhedral surface of a cow. A data file for the "cow" can be found in the CGAL library in the folder:
`cgal/Surface_mesh_shortest_path/test/Surface_mesh_shortest_path/data/`

B.3 Cyclide

The cyclide is a geometric inversion of the torus. In implicitly form, it is given by:

$$x^2+y^2+z^2-\mu^2+b^2)^2 - 4(ax-c\mu)^2 - 4b^2y^2 = 0.$$

We take $a = 7$, $b = \sqrt{48}$, $c = \sqrt{a^2-b^2}$ and $\mu = 3$. Note that if $a = b$, we obtain a standard torus.

B.4 Dino

The "Dino" surface is a polyhedral surface of a brachiosaurus that is characterized by cylinder-like parts (long neck and legs). A data file for the "Dino" can be found in the CGAL library in the folder:
`cgal/Surface_mesh_segmentation/benchmark/Surface_mesh_segmentation/data/`

B.5 Elephant

"Elephant" is a polyhedral surface of an elephant that is particularly interesting from a topological point of view as it possesses many thin regions and holes. A data file for the "Elephant" can be found in the CGAL library in the folder:
`cgal/Surface_mesh/examples/Surface_mesh/data/`

B.6 Ellipsoid

The equation of an ellipsoid is:

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2} = 1.$$

where a, b and c are the lengths of the semi axis of the ellipsoid.

B.7 Fandisk

"Fandisk" is a polyhedral surface of a mechanical piece characterized by large flat surfaces and sharp edges. A data file for the "Fandisk" can be found in the CGAL library in the folder:
`cgal/Surface_mesh_shortest_path/benchmark/Surface_mesh_shortest_path/data/`

B.8 Fertility

"Fertility" is a polyhedral surface of a statue that is overall very smooth with various cylinder-like parts. It is taken from the AIM@SHAPE repository.

B.9 Oni

The "Oni" surface is an open polyhedral (sur)face. A data file for the "Oni" surface can be found in the CGAL library in the folder:
`cgal/Surface_mesh_parameterization/test/Surface_mesh_parameterization/data`

Bibliography

[1] NASA Advanced Supercomputing Division. `http://www.nas.nasa.gov/SC11/demos/demo11.html`.

[2] CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.

[3] *The Boost Graph Library: User Guide and Reference Manual*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.

[4] Alauzet, F., and Loseille, A. A decade of progress on anisotropic mesh adaptation for computational fluid dynamics. *Computer-Aided Design 72* (2016), 13–39.

[5] Alliez, P., Cohen-Steiner, D., Devillers, O., Lévy, B., and Desbrun, M. Anisotropic polygonal remeshing. In *ACM Transactions on Graphics (TOG)* (2003), vol. 22, ACM, pp. 485–493.

[6] Alliez, P., Cohen-Steiner, D., Yvinec, M., and Desbrun, M. Variational tetrahedral meshing. *ACM Trans. Graph. 24*, 3 (2005), 617–625.

[7] Alliez, P., Tayeb, S., and Wormser, C. 3D fast intersection and distance computation (AABB tree). *CGAL User and Reference Manual* (2012).

[8] Amenta, N., and Bern, M. Surface reconstruction by Voronoi filtering. *Discrete and Computational Geometry 22*, 4 (1999), 481–504.

[9] Amenta, N., and Bern, M. Surface reconstruction by Voronoifiltering. *Discrete Comput. Geom. 22*, 4 (1999), 481–504.

[10] Arsigny, V., Fillard, P., Pennec, X., and Ayache, N. Log-Euclidean metrics for fast and simple calculus on diffusion tensors. *Magnetic resonance in medicine 56*, 2 (2006), 411–421.

[11] Aurenhammer, F. Power Diagrams: Properties, Algorithms and Applications. *SIAM J. Comput. 16*, 1 (1987), 78–96.

[12] Aurenhammer, F., and Klein, R. Voronoi diagrams. In *Handbook of Computational Geometry*, J. Sack and G. Urrutia, Eds. Elsevier Science Publishing, 2000, pp. 201–290.

[13] Batchelor, P., Moakher, M., Atkinson, D., Calamante, F., and Connelly, A. A rigorous framework for diffusion tensor calculus. *Magnetic Resonance in Medicine 53*, 1 (2005), 221–225.

[14] Bentley, J. L. Multidimensional binary search trees used for associative searching. *Communications of the ACM 18*, 9 (1975), 509–517.

[15] Boissonnat, J.-D., Chazal, F., and Yvinec, M. *Geometry and Topology Inference*. in preparation.

[16] Boissonnat, J.-D., Cohen-Steiner, D., and Yvinec, M. Comparison of algorithms for anisotropic meshing and adaptive refinement. Research Report ACS-TR-362603, INRIA, 2008.

[17] Boissonnat, J.-D., Dyer, R., and Ghosh, A. Constructing intrinsic Delaunay triangulations of submanifolds. Research Report RR-8273, INRIA, 2013. arXiv:1303.6493.

[18] Boissonnat, J.-D., Dyer, R., and Ghosh, A. Delaunay stability via perturbations. Research Report RR-8275, INRIA, 2013.

[19] Boissonnat, J.-D., Dyer, R., and Ghosh, A. The stability of Delaunay triangulations. *Int. J. Comp. Geom. & Appl. 23*, 04n05 (2013), 303–333. eprint: arXiv:1304.2947.

[20] Boissonnat, J.-D., Dyer, R., and Ghosh, A. Delaunay stability via perturbations. *Int. J. Comp. Geom. & Appl. to appear* (2014). (Preprint: arXiv:1310.7696).

[21] Boissonnat, J.-D., Dyer, R., Ghosh, A., and Oudot, S. Equating the witness and restricted Delaunay complexes. Tech. Rep. CGL-TR-24, Computational Geometric Learning, 2011. http://cgl.uni-jena.de/Publications/WebHome.

[22] Boissonnat, J.-D., Dyer, R., Ghosh, A., and Oudot, S. Y. Only distances are required to reconstruct submanifolds. Research report, INRIA Sophia Antipolis, 2014.

[23] Boissonnat, J.-D., and Flötotto, J. A coordinate system associated with points scattered on a surface. *Computer-Aided Design 36*, 2 (2004), 161 – 174. Solid Modeling and Applications.

[24] Boissonnat, J.-D., and Ghosh, A. Triangulating smooth submanifolds with light scaffolding. *Mathematics in Computer Science 4*, 4 (2010), 431–461.

[25] Boissonnat, J.-D., and Ghosh, A. Manifold reconstruction using tangential Delaunay complexes. *Discrete and computational Geometry*, November (2013).

[26] Boissonnat, J.-D., Guibas, L. J., and Oudot, S. Y. Manifold reconstruction in arbitrary dimensions using witness complexes. *Discrete & Comp. Geom. 42* (2009), 37–70.

[27] Boissonnat, J.-D., and Oudot, S. Provably good sampling and meshing of surfaces. *Graphical Models 67*, 5 (2005), 405–451.

[28] Boissonnat, J.-D., Shi, K.-L., Tournois, J., and Yvinec, M. Anisotropic Delaunay meshes of surfaces. *ACM Trans. Graph. 34*, 2 (2015), 14:1–14:11.

[29] Boissonnat, J.-D., Wormser, C., and Yvinec, M. Anisotropic diagrams: Labelle Shewchuk approach revisited. *Theoret. Comput. Sci. 408* (2008), 163–173.

[30] Boissonnat, J.-D., Wormser, C., and Yvinec, M. Locally uniform anisotropic meshing. In *Proceedings of the twenty-fourth annual symposium on Computational geometry* (2008), ACM, pp. 270–277.

[31] Boissonnat, J.-D., Wormser, C., and Yvinec, M. Anisotropic Delaunay mesh generation. *SIAM Journal on Computing 44*, 2 (2015), 467–512.

[32] Boissonnat, J.-D., and Yvinec, M. *Algorithmic geometry*. Cambridge University Press, 1998.

[33] Borouchaki, H., George, P. L., Hecht, F., Laug, P., and Saltel, E. Delaunay mesh generation governed by metric specifications. part Ialgorithms. *Finite Elem. Anal. Des. 25*, 1-2 (1997), 61–83.

[34] Bowyer, A. Computing dirichlet tessellations. *The Computer Journal 24*, 2 (1981), 162–166.

[35] Busé, L., Khalil, H., and Mourrain, B. *Resultant-Based Methods for Plane Curves Intersection Problems*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 75–92.

[36] Campen, M., Heistermann, M., and Kobbelt, L. Practical anisotropic geodesy. In *Proceedings of the Eleventh Eurographics/ACMSIGGRAPH Symposium on Geometry Processing* (2013), SGP '13, Eurographics Association, pp. 63–71.

[37] Canas, G.-D., and Gortler, S.-J. Orphan-free anisotropic Voronoi diagrams. *Discrete and Computational Geometry 46*, 3 (2011).

[38] Cañas, G. D., and Gortler, S. J. Duals of orphan-free anisotropic Voronoi diagrams are embedded meshes. In *SoCG* (New York, NY, USA, 2012), ACM, pp. 219–228.

[39] Cao, T.-t., Edelsbrunner, H., and Tan, T.-s. Proof of correctness of the digital Delaunay triangulation algorithm. *Computational Geometry: Theory and Applications 48* (2015).

[40] Cazals, F., and Pouget, M. Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design 22*, 2 (2005), 121–146.

[41] Chavel, I. *Riemannian Geometry, A modern introduction*, 2nd ed. Cambridge, 2006.

[42] Chen, L. Mesh smoothing schemes based on optimal Delaunay triangulations. In *Proceedings of the 13th International Meshing Roundtable* (2004), Sandia National Laboratories, pp. 109–120.

[43] Chen, L., Sun, P., and Xu, J. Optimal anisotropic meshes for minimizing interpolation errors in L^p-norm. *Mathematics of Computation 76* (January 2007), 179–204.

[44] Chen, L., and Xu, J. Optimal Delaunay triangulations. *Journal of Computational Mathematics 22* (2004), 299–308.

[45] Cheng, H.-L., Dey, T. K., Edelsbrunner, H., and Sullivan, J. Dynamic skin triangulation. *Discrete & Computational Geometry 25*, 4 (2001), 525–568.

[46] Cheng, S., Dey, T., and Levine, J. Theory of a practical Delaunay meshing algorithm for a large class of domains. *Algorithms, Architecture and Information Systems Security, B. Bhattacharya, S. Sur-Kolay, S. Nandy, and A. Bagchi, Eds 3* (2008), 17–41.

[47] Cheng, S. W., Dey, T., and Levine, J. A practical Delaunay meshing algorithm for a large class of domains*. In *Proceedings of the 16th International Meshing Roundtable* (2008), Springer, pp. 477–494.

[48] Cheng, S.-W., and Dey, T. K. Quality meshing with weighted Delaunay refinement. In *SODA '02: Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms* (Philadelphia, PA, USA, 2002), Society for Industrial and Applied Mathematics, pp. 137–146.

[49] Cheng, S.-W., Dey, T. K., Edelsbrunner, H., Facello, M. A., and Teng, S.-H. Silver exudation. *J. ACM 47*, 5 (2000), 883–904.

[50] Cheng, S.-W., Dey, T. K., Ramos, E. A., and Wenger, R. Anisotropic surface meshing. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm* (2006), Society for Industrial and Applied Mathematics, pp. 202–211.

[51] Chevalier, C., and Pellegrini, F. Pt-scotch: A tool for efficient parallel graph ordering. *Parallel computing 34*, 6 (2008), 318–331.

[52] Chew, L. P. Constrained Delaunay triangulations. *Algorithmica 4*, 1 (1989), 97–108.

[53] Chew, L. P. Guaranteed-quality mesh generation for curved surfaces. In *SCG '93: Proceedings of the ninth annual symposium on Computational geometry* (1993), ACM Press, pp. 274–280.

[54] Coplien, J. O. Curiously recurring template patterns. *C++ Report 7*, 2 (1995), 24–27.

[55] Crane, K., Weischedel, C., and Wardetzky, M. Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Trans. Graph. 32*, 5 (2013), 152:1–152:11.

[56] D'Azevedo, E. F. Are bilinear quadrilaterals better than linear triangles? *SIAM Journal on Scientific Computing 22*, 1 (2000), 198–217.

[57] D'Azevedo, E. F., and Simpson, R. B. On optimal interpolation triangle incidences. *SIAM J. Sci. Statist. Comput. 10*, 6 (1989), 1063–1075.

[58] Devillers, O. The Delaunay hierarchy. *International Journal of Foundations of Computer Science 13*, 02 (2002), 163–180.

[59] Diaz, M. C., Hecht, F., Mohammadi, B., and Pironneau, O. Anisotropic unstructured mesh adaptation for flows simulations. *Internat. J. Numer. Methods Fluids 25* (1997), 475–491.

[60] Dobrzynski, C., and Frey, P. Anisotropic Delaunay mesh adaptation for unsteady simulations. *Proceedings of the 17th International Meshing Roundtable* (2008), 177–194.

[61] Dobrzynski, C., Melchior, M., Delannay, L., and Remacle, J.-F. A mesh adaptation procedure for periodic domains. *International journal for numerical methods in engineering 86*, 12 (2011), 1396–1412.

[62] Du, Q., Faber, V., and Gunzburger, M. Centroidal Voronoi tessellations: applications and algorithms. *SIAM review 41*, 4 (1999), 637–676.

[63] Du, Q., and Wang, D. Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations. *International journal for numerical methods in engineering 56*, 9 (2003), 1355–1373.

[64] Du, Q., and Wang, D. Anisotropic centroidal Voronoi tessellations and their applications. *SIAM Journal on Scientific Computing 26*, 3 (2005), 737–761.

[65] Dyer, R., Vegter, G., and Wintraecken, M. Riemannian simplices and triangulations. Preprint: arXiv:1406.3740.

[66] Dyer, R., Zhang, H., and Möller, T. Surface sampling and the intrinsic Voronoi diagram. *Computer Graphics Forum (Special Issue of Symp. Geometry Processing) 27*, 5 (2008), 1393–1402.

[67] Dyer, R., Zhang, H., Möller, T., and Clements, A. An investigation of the spectral robustness of mesh Laplacians. Tech. Rep. TR 2007-17, Simon Fraser University, 2007. SFU-CMPT.

[68] Edelsbrunner, H., and Shah, N. R. Triangulating topological spaces. In *SoCG* (1994), pp. 285–292.

[69] Edelsbrunner, H., and Shah, N. R. Triangulating topological spaces. *Int. J. Comput. Geometry Appl. 7*, 4 (1997), 365–378.

[70] Flötotto, J. *A Coordinate System associated to a Point Cloud issued from a Manifold: Definition, Properties and Applications.* Theses, Université Nice Sophia Antipolis, 2003.

[71] Frey, P., and George, P. L. *Mesh generation: Application to finite elements.* Hermes Science, 2008.

[72] Frey, P., and George, P.-L. *Mesh generation.* John Wiley & Sons, 2013.

[73] Fu, X.-M., Liu, Y., Snyder, J., and Guo, B. Anisotropic simplicial meshing using local convex functions. *ACM Trans. Graph. 33*, 6 (2014), 182:1–182:11.

[74] Funke, S., Klein, C., Mehlhorn, K., and Schmitt, S. Controlled perturbation for delaunay triangulations. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms* (2005), Society for Industrial and Applied Mathematics, pp. 1047–1056.

[75] Garland, M., and Heckbert, P. S. Surface simplification using quadric error metrics. In *ACM SIGGRAPH* (1997), pp. 209–216.

[76] Hecht, F., and Mohammadi, B. Mesh adaption by metric control for multi-scale phenomena and turbulence. In *AIAA 35th Aerospace Sciences Meeting & Exhibit* (1997).

[77] Helander, P., Beidler, C., Bird, T., Drevlak, M., Feng, Y., Hatzky, R., Jenko, F., Kleiber, R., Proll, J., Turkin, Y., et al. Stellarator and tokamak plasmas: a comparison. *Plasma Physics and Controlled Fusion 54*, 12 (2012), 124009.

[78] Hoff III, K. E., Keyser, J., Lin, M., Manocha, D., and Culver, T. Fast computation of generalized Voronoi diagrams using graphics hardware. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (1999), ACM Press/Addison-Wesley Publishing Co., pp. 277–286.

[79] Hughes, J. F. Differential geometry of implicit surfaces in 3-space-a primer. Research report, Brown University Providence, Rhode Island, 2003.

[80] Jamin, C., Alliez, P., Yvinec, M., and Boissonnat, J.-D. Cgalmesh: a generic framework for Delaunay mesh generation. *ACM Transactions on Mathematical Software (TOMS) 41*, 4 (2015), 23.

[81] Jason Wolff, P. University of Minnesota Twin Cities.

[82] Jiao, X., Colombi, A., Ni, X., and Hart, J. Anisotropic mesh adaptation for evolving triangulated surfaces. *Engineering with Computers 26*, 4 (2010), 363–376.

[83] Karcher, H. Riemannian center of mass and mollifier smoothing. *Communications on Pure and Applied Mathematics 30* (1977), 509–541.

[84] Karypis, G., and Kumar, V. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing 20*, 1 (1998), 359–392.

[85] Kendall, W. Probability, convexity, and harmonic maps with small image I: Uniqueness and fine existence. *Procedings of the London Mathematical society s3-61 (Issue 2)* (1990), 371–406.

[86] Kindlmann, G., Estepar, R. S. J., Niethammer, M., Haker, S., and Westin, C.-F. Geodesic-loxodromes for diffusion tensor interpolation and difference measurement. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2007), Springer, pp. 1–9.

[87] Klein, P. P. On the ellipsoid and plane intersection equation. *Applied Mathematics 3*, 11 (2012), 1634.

[88] Konukoglu, E., Sermesant, M., Clatz, O., Peyrat, J.-M., Delingette, H., and Ayache, N. A recursive anisotropic fast marching approach to reaction diffusion equation: Application to tumor growth modeling. In *Proceedings of the 20th International Conference* (2007), pp. 687–699.

[89] Labelle, F., and Shewchuk, J. R. Anisotropic Voronoi diagrams and guaranteed-quality anisotropic mesh generation. In *SCG' 03 : Proceedings of the nineteenth annual symposium on Computational geometry* (New York, NY, USA, 2003), ACM Press, pp. 191–200.

[90] Leibon, G. *Random Delaunay triangulations, the Thurston-Andreev theorem, and metric uniformization*. PhD thesis, UCSD, 1999. arXiv:math/0011016v1.

[91] Leibon, G., and Letscher, D. Delaunay triangulations and Voronoi diagrams for Riemannian manifolds. In *SoCG* (2000), pp. 341–349.

[92] Lévy, B., and Bonneel, N. Variational anisotropic surface meshing with Voronoi parallel linear enumeration. In *Proceedings of the 21st international meshing roundtable*. Springer, 2013, pp. 349–366.

[93] Lévy, B., and Liu, Y. L_p centroidal Voronoi Tessellation and its applications. In *ACM Transactions on Graphics (TOG)* (2010), vol. 29, ACM, p. 119.

[94] Li, X.-Y. *Sliver-free Three Dimensional Delaunay Mesh Generation*. PhD thesis, University of Illinois at Urbana Champaign, PhD thesis, University of Illinois at Urbana-Champaign 2000.

[95] Li, X.-Y. Generating well-shaped d-dimensional Delaunay meshes. *Theor. Comput. Sci. 296*, 1 (2003), 145–165.

[96] Li, X.-Y., and Teng, S.-H. Generating well-shaped Delaunay meshed in 3d. In *SODA '01: Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms* (2001), Society for Industrial and Applied Mathematics, pp. 28–37.

[97] Liu, Y., Pan, H., Snyder, J., Wang, W., and Guo, B. Computing self-supporting surfaces by regular triangulation. *ACM Trans. Graph. 32*, 4 (2013), 92:1–92:10.

[98] Liu, Y., Wang, W., Lévy, B., Sun, F., Yan, D.-M., Lu, L., and Yang, C. On centroidal Voronoi tessellation, energy smoothness and fast computation. *ACM Trans. Graph. 28*, 4 (2009), 101:1–101:17.

[99] Liu, Y.-J., Xu, C., He, Y., and Kim, D.-S. The duality of geodesic Voronoi / Delaunay diagrams for an intrinsic discrete Laplace-Beltrami operator on simplicial surfaces. In *CCCG* (2014), Citeseer.

[100] Lloyd, S. Least squares quantization in pcm. *IEEE Trans. Inf. Theor. 28*, 2 (2006), 129–137.

[101] Loseille, A. Metric-orthogonal anisotropic mesh generation. *Procedia Engineering 82* (2014), 403–415.

[102] Mavriplis, D. J. Adaptive mesh generation for viscous flows using triangulation. *Journal of computational Physics 90*, 2 (1990), 271–291.

[103] Mavriplis, D. J. Unstructured mesh generation and adaptivity. Tech. rep., DTIC Document, 1995.

[104] Miller, G. L., Phillips, T., and Sheehy, D. R. Linear-size meshes. In *20th Canadian Conference on Computational Geometry* (Montreal, August 2008).

[105] Mirebeau, J.-M. Anisotropic fast-marching on cartesian grids using lattice basis reduction. *SIAM Journal on Numerical Analysis 52*, 4 (2014).

[106] Mullen, P., Memari, P., de Goes, F., and Desbrun, M. Hot: Hodge-optimized triangulations. *ACM Trans. Graph. 30*, 4 (2011), 103:1–103:12.

[107] Nadler, E. Piecewise linear best l2 approximation on triangulations. *Approximation Theory V* (1986), 499–502.

[108] Nass, H., Wolter, F.-E., Thielhelm, H., and Dogan, C. Computation of geodesic Voronoi diagrams in riemannian 3-space using medial equations. In *Cyberworlds, 2007. CW'07. International Conference on* (2007), IEEE, pp. 376–385.

[109] Nintendo. The Legend of Zelda: The Wind Waker. GameCube, 2002.

[110] Nivoliers, V., Lévy, B., and Geuzaine, C. Anisotropic and feature sensitive triangular remeshing using normal lifting. *Journal of Computational and Applied Mathematics 289* (2015), 225–240.

[111] Niyogi, P., Smale, S., and Weinberger, S. Finding the homology of submanifolds with high confidence from random samples. *Discrete & Comp. Geom. 39*, 1-3 (2008), 419–441.

[112] Okabe, A., Boots, B., Sugihara, K., and Chiu, S. N. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, second ed. John Wiley, 2000.

[113] Owen, S. J. A survey of unstructured mesh generation technology. In *IMR* (1998), pp. 239–267.

[114] Panozzo, D., Puppo, E., Tarini, M., and Sorkine-Hornung, O. Frame fields: anisotropic and non-orthogonal cross fields. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 134.

[115] Peyré, G., and Cohen, L. Geodesic remeshing using front propagation. *International Journal on Computer Vision 69*, 1 (2006), 145–156.

[116] Peyré, G., Péchaud, M., Keriven, R., and Cohen, L. D. Geodesic methods in computer vision and graphics. *Found. Trends. Comput. Graph. Vis. 5* (2010), 197–397.

[117] Preparata, F. P., and Shamos, M. *Computational geometry: an introduction*. Springer Science & Business Media, 2012.

[118] Rineau, L. *Meshing Volumes Bounded by Piecewise Smooth Surfaces*. PhD thesis, Université Paris Diderot - Paris 7, 2007.

[119] Rineau, L., and Yvinec, M. A generic software design for Delaunay refinement meshing. *Computational Geometry 38*, 1 (2007), 100–110.

[120] Rineau, L., and Yvinec, M. Meshing 3Ddomains bounded by piecewise smooth surfaces. In *Meshing Roundtable conference proceedings* (2007), pp. 443–460.

[121] Rourke, C., and Sanderson, B. *Introduction to piecewise-linear topology*. Springer Science & Business Media, 2012.

[122] Ruppert, J. A new and simple algorithm for quality 2-dimensional mesh generation. In *SODA* (1993), pp. 83–92.

[123] Ruppert, J. A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *J. of Algorithms 18*, 3 (1995), 548–585.

[124] Rustamov, R. Barycentric coordinates on surfaces. *Eurographics Symposium on Geometry Processing 29*, 5 (2010).

[125] Schmidt, R., Grimm, C., and Wyvill, B. Interactive decal compositing with discrete exponential maps. *ACM Transactions on Graphics (TOG) 25*, 3 (2006), 605–613.

[126] Schoen, J. Robust, guaranteed-quality anisotropic mesh generation. M.S. thesis, University of California ar Berkeley, 2008.

[127] Shewchuk, J. R. What is a good linear finite element? Interpolation, conditioning, anisotropy, and quality measures. In *http://www.cs.cmu.edu/~jrs/jrspapers.html* (Manuscript 2002).

[128] Shewchuk, R. Star splaying: an algorithm for repairing Delaunay triangulations and convex hulls. In *SCG '05: Proceedings of the twenty-first annual symposium on Computational geometry* (New York, NY, USA, 2005), ACM, pp. 237–246.

[129] Shimada, K., and Gossard, D. C. Bubble mesh: automated triangular meshing of non-manifold geometry by sphere packing. In *Proceedings of the third ACM symposium on Solid modeling and applications* (1995), ACM, pp. 409–419.

[130] Shimada, K., Yamada, A., and Itoh, T. Anisotropic triangulation of parametric surfaces via close packing of ellipsoids. *International Journal of Computational Geometry & Applications* (200).

[131] Sperner, E. Fifty years of further development of a combinatorial lemma. *Numerical solution of highly nonlinear problems (Sympos. Fixed Point Algorithms and Complementarity Problems, Univ. Southampton, Southampton), Part A* (1980), 183–197.

[132] Tournois, J. *Optimisation de maillages*. PhD thesis, Université Nice Sophia Antipolis, 2009.

[133] Tournois, J., Srinivasan, R., and Alliez, P. Perturbing slivers in 3D Delaunay meshes. In *Proceedings of the 18th International Meshing Roundtable*. Springer, 2009, pp. 157–173.

[134] Tournois, J., Wormser, C., Alliez, P., and Desbrun, M. Interleaving Delaunay refinement and optimization for practical isotropic tetrahedron mesh generation. *ACM Trans. Graph. 28*, 3 (2009), 75:1–75:9.

[135] Vallet, M., Hecht, F., and Mantel, B. Anisotropic control of mesh generation based upon a voronoi type method. *Numerical grid generation in computational fluid dynamics and related fields* (1991), 93–103.

[136] Vallet, M.-G., Dompierre, J., Bourgault, Y., Fortin, M., and Habashi, W. G. Coupling flow solvers and grids through an edge-based adaptive grid method. *Fluids Eng. Conf. 238* (1996), 209–216.

[137] Wang, X., Ying, X., Liu, Y.-J., Xin, S.-Q., Wang, W., Gu, X., Mueller-Wittig, W., and He, Y. Intrinsic computation of centroidal Voronoi tessellation (CVT)on meshes. *Computer-Aided Design 58* (2015), 51–61.

[138] Watson, D. F. Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes. *The computer journal 24*, 2 (1981), 167–172.

[139] Whitney, H. *Geometric Integration Theory*. Princeton University Press, 1957.

[140] Wintraecken, M. *Ambient and Intrinsic Triangulations and Topological Methods in Cosmology*. PhD thesis, Rijksuniversiteit Groningen, 9 2015.

[141] Xu, C., Liu, Y.-J., Sun, Q., Li, J., and He, Y. Polyline-sourced geodesic Voronoi diagrams on triangle meshes. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 161–170.

[142] Yamakawa, S., and Shimada, K. Anisotropic tetrahedral meshing via bubble packing and advancing front. *International Journal for Numerical Methods in Engineering 57*, 13 (2003), 1923–1942.

[143] Yoo, S. W., Seong, J.-K., Sung, M.-H., Shin, S. Y., and Cohen, E. A triangulation-invariant method for anisotropic geodesic map computation on surface meshes. *Visualization and Computer Graphics, IEEE Transactions on 18*, 10 (2012), 1664–1677.

[144] Zhong, Z., Guo, X., Wang, W., Lévy, B., Sun, F., Liu, Y., and Mao, W. Particle-based anisotropic surface meshing. *ACM Trans. Graph. 32*, 4 (2013), 99:1–99:14.

[145] Zhong, Z., Shuai, L., Jin, M., and Guo, X. Anisotropic surface meshing with conformal embedding. *Graph. Models 76*, 5 (2014), 468–483.