



Anisotropic neighborhoods of superpixels for thin structure segmentation

Christophe Ribal

► To cite this version:

Christophe Ribal. Anisotropic neighborhoods of superpixels for thin structure segmentation. Computer Vision and Pattern Recognition [cs.CV]. Université Paris-Saclay, 2021. English. NNT : 2021UP-ASG117 . tel-03606136

HAL Id: tel-03606136

<https://theses.hal.science/tel-03606136>

Submitted on 11 Mar 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Anisotropic neighborhoods of superpixels for thin structure segmentation

*Voisinages anisotropes de superpixels pour la segmentation d'images
de structures fines*

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 580: Sciences et Technologies de
l'Information et de la Communication (STIC)

Spécialité de doctorat: Traitement du signal et des images

Graduate School : GS Informatique et Sciences du Numérique

Référent : Faculté des sciences d'Orsay

Thèse préparée dans l'unité de recherche Université Paris-Saclay, ENS Paris-Saclay,
CNRS, SATIE, 91190, Gif-Sur-Yvettes, France, sous la direction de Sylvie LE
HÉGARAT-MASCLE, Professeure des Universités à l'Université Paris Sud, et le
co-encadrement de Nicolas LERMÉ, Maître de Conférence au Laboratoire SATIE.

Thèse soutenue à Paris-Saclay, le 17 décembre 2021, par

Christophe RIBAL

Composition du jury

M. Thomas Rodet,
Pr. Université ENS Paris Saclay
Mme Sylvie Chambon,
MCF INP Toulouse
M. Nicolas Passat,
Pr. Université Reims
Mme Su Ruan,
Pr. Université Rouen
Mme Sylvie Le Hégarat-Masclé,
Pr. Université Paris Sud
M. Nicolas Lermé,
MCF Laboratoire SATIE

Président
Rapporteur & Examinatrice
Rapporteur & Examineur
Examinatrice
Directrice de thèse
Co-encadrant & Examineur

Contents

| | |
|--|------------|
| Contents | iii |
| List of figures | vii |
| List of Tables | ix |
| 1 Introduction | 1 |
| 2 Mathematical modeling | 5 |
| 2.1 Background | 6 |
| 2.1.1 Regularization and inverse problems | 6 |
| 2.1.2 The notion of neighborhood | 7 |
| 2.1.3 Superpixel segmentation | 9 |
| 2.2 Thin structures | 11 |
| 2.2.1 Definition | 11 |
| 2.2.2 Thin structure detection | 11 |
| 2.2.3 Shape From Focus (SFF) | 12 |
| 2.3 Problem formulation | 14 |
| 2.3.1 Markov Random Fields (MRF) | 14 |
| 2.3.2 Energy functional | 16 |
| 2.3.3 2D to 3D modeling | 17 |
| 2.3.4 Proposed approach | 18 |
| 3 Neighborhood Construction | 19 |
| 3.1 Isotropic Neighborhood | 20 |
| 3.1.1 Stawiasky's neighborhood | 21 |
| 3.1.2 Disc neighborhood | 21 |
| 3.2 Thin Structures Estimation | 22 |
| 3.2.1 Energy based guidance map | 23 |
| 3.2.2 Tensor Voting | 26 |
| 3.2.3 RORPO | 32 |
| 3.3 Anisotropic Neighborhoods | 36 |
| 3.3.1 Shape-based neighborhood | 36 |
| 3.3.2 Dictionary-based neighborhood | 38 |
| 3.3.3 Path-based neighborhood | 39 |
| 3.4 Conclusion | 41 |
| 4 Numerical resolution | 43 |
| 4.1 Optimization with graph cuts | 44 |
| 4.1.1 Definitions | 44 |
| 4.1.2 Energy minimization and max-flow algorithms | 47 |
| 4.1.3 From binary to multi-labels problem resolution | 53 |
| 4.2 Functional minimization | 59 |
| 4.2.1 Dictionary | 59 |

| | | |
|----------|---|------------|
| 4.2.2 | TV and RORPO based neighborhood | 60 |
| 4.2.3 | Direct neighborhood estimation | 60 |
| 4.3 | Implementation | 61 |
| 4.3.1 | Supapixel generation | 61 |
| 4.3.2 | Programming | 62 |
| 4.3.3 | Parameters setting | 65 |
| 5 | Thin structure detection | 69 |
| 5.1 | Context | 70 |
| 5.2 | Binary implementation | 70 |
| 5.2.1 | Energy terms | 70 |
| 5.2.2 | Parameter tuning | 71 |
| 5.3 | Numerical experiments | 72 |
| 5.3.1 | Guidance map estimation | 72 |
| 5.3.2 | Qualitative and quantitative evaluation | 72 |
| 5.4 | Conclusions | 76 |
| 6 | Shape from focus (SFF) | 77 |
| 6.1 | Problem and state of the art | 78 |
| 6.1.1 | Principle | 78 |
| 6.1.2 | Geometrical optics | 78 |
| 6.1.3 | From physics to sharpness operators | 79 |
| 6.1.4 | Depth derivation from sharpness | 79 |
| 6.1.5 | SFF in variational framework | 81 |
| 6.2 | Model application to SFF | 81 |
| 6.2.1 | Data in SFF | 81 |
| 6.2.2 | Anisotropic regularization | 82 |
| 6.2.3 | Thin structure estimation in SFF | 83 |
| 6.3 | Proposed evaluation | 84 |
| 6.3.1 | Depth error map | 84 |
| 6.3.2 | Neighborhood quality estimation | 86 |
| 6.3.3 | Global metrics | 93 |
| 6.4 | Experimental results | 94 |
| 6.4.1 | Global segmentation results | 94 |
| 6.4.2 | Sensitivity to the regularization parameter | 116 |
| 6.4.3 | Result tables | 116 |
| 6.5 | Conclusion | 127 |
| 7 | Conclusions and perspectives | 129 |
| 7.1 | Conclusions | 129 |
| 7.2 | Perspectives | 130 |
| A | Synthèse en Français | I |
| A.1 | Introduction générale | I |
| A.2 | Voisinages anisotropes | III |
| A.2.1 | Estimation de l'orientation | IV |
| A.2.2 | Construction du voisinage | IV |
| A.3 | Applications et conclusions | VI |
| B | Demonstrations | IX |
| B.1 | Energy based guidance map | IX |
| B.1.1 | Pseudometric | IX |
| B.2 | Tensor Voting | XII |
| B.2.1 | Geometry and stick kernel | XII |

| | |
|--|---------------|
| C Submitted article | XV |
| C.1 Introduction | XV |
| C.2 Superpixels-based SFF | XVI |
| C.2.1 Basics of SFF | XVI |
| C.2.2 From pixel level to superpixel one | XVI |
| C.2.3 Energetic formulation | XVII |
| C.2.4 Optimization | XVIII |
| C.3 Anisotropic neighborhood construction | XVIII |
| C.3.1 Tensor Voting-based guidance map | XVIII |
| C.3.2 RORPO-based guidance map | XIX |
| C.3.3 Path-based neighborhoods | XXI |
| C.4 Experiments and results | XXII |
| C.4.1 Data | XXII |
| C.4.2 Evaluation criteria | XXIII |
| C.4.3 Alternative approaches considered for comparison | XXIII |
| C.4.4 Results | XXIV |
| C.5 Conclusion and perspectives | XXX |
| C.6 3D Tensor Voting | XXX |
| D References | XXXI |
| E Acronyms list | XXXVII |
| F Symbols list | XXXIX |

List of figures

| | | |
|------|---|----|
| 2.1 | Image processing - Illustration | 5 |
| 2.2 | Gradient descent algorithm | 7 |
| 2.3 | The benefits of regularization | 8 |
| 2.4 | Usual neighborhoods | 9 |
| 2.5 | Thin structures examples | 11 |
| 2.6 | Data in crack detection | 12 |
| 2.7 | Data for SFF | 13 |
| 2.8 | Example of cliques | 15 |
| 2.9 | Flowchart of the proposed approach | 18 |
| 3.1 | Stawiasky's isotropic neighborhood | 21 |
| 3.2 | Disc isotropic neighborhood | 22 |
| 3.3 | Energy based vesselness | 24 |
| 3.4 | Energy term E_3^{st} | 26 |
| 3.5 | The Tensor Voting framework | 27 |
| 3.6 | Tensor decomposition | 28 |
| 3.7 | Stick vote notations | 29 |
| 3.8 | The RORPO framework | 32 |
| 3.9 | Orientation sampling | 33 |
| 3.10 | Path opening | 34 |
| 3.11 | Behavior of Ranking Filter | 35 |
| 3.12 | Shape-based neighborhood | 37 |
| 3.13 | Dictionary neighborhood | 38 |
| 3.14 | Target-based neighborhood | 40 |
| 3.15 | Cardinal-based neighborhood | 41 |
| 4.1 | Numerical Resolution - Illustration | 43 |
| 4.2 | An example of $s - t$ cut | 45 |
| 4.3 | Graph construction for \mathcal{F}^1 | 48 |
| 4.4 | Graph construction for \mathcal{F}^2 | 49 |
| 4.5 | Max-flow algorithm illustration | 50 |
| 4.6 | Boykov max-flow algorithm | 52 |
| 4.7 | Example of multi label graphs | 53 |
| 4.8 | α -expansion and α - β swap moves | 54 |
| 4.9 | Comparison of superpixels algorithms | 63 |
| 4.10 | ETPS superpixel construction | 64 |
| 4.11 | FH, ETPS and WP on Art image | 64 |
| 4.12 | Graphical User Interface | 66 |
| 5.1 | Energy-based and TV-based guidance map comparison | 73 |
| 5.2 | Crack segmentation examples at pixel level | 73 |
| 5.3 | Evaluation of superpixel binary segmentation performances on examples | 75 |

| | | |
|------|--|--------|
| 6.1 | SFF illustration | 78 |
| 6.2 | Image formation model | 80 |
| 6.3 | SFF depth reconstruction | 80 |
| 6.4 | Brush scene ground truth | 85 |
| 6.5 | Brush scene depth error map | 86 |
| 6.6 | Brush scene segmentation map | 87 |
| 6.7 | Brush Sample Segmentation Map | 88 |
| 6.8 | Brush scene error map histogram | 89 |
| 6.9 | Neighborhood qualitative criteria | 92 |
| 6.10 | Global SFF results | 95 |
| 6.11 | Reindeer scene | 96 |
| 6.12 | Reindeer superpixels | 97 |
| 6.13 | Reindeer depth maps | 99 |
| 6.14 | Reindeer error maps | 100 |
| 6.15 | Reindeer error map boxplot | 101 |
| 6.16 | Reindeer neighborhood criteria | 101 |
| 6.17 | Lampshade scene | 102 |
| 6.18 | Lampshade depth maps | 103 |
| 6.19 | Lampshade error maps | 104 |
| 6.20 | Lampshade error map boxplot | 105 |
| 6.21 | Lampshade neighborhood criteria | 105 |
| 6.22 | Plastic scene | 106 |
| 6.23 | Plastic depth maps | 108 |
| 6.24 | Plastic error maps | 109 |
| 6.25 | Plastic neighborhood criteria | 110 |
| 6.26 | Bowling scene | 111 |
| 6.27 | Bowling depth maps | 112 |
| 6.28 | Bowling error maps | 113 |
| 6.29 | Bowling neighborhood criteria | 114 |
| 6.30 | Art scene | 116 |
| 6.31 | Art depth maps | 117 |
| 6.32 | Lampshade error maps | 118 |
| 6.33 | Art neighborhood criteria | 119 |
| 6.34 | RMSE at superpixel level | 120 |
| 6.35 | RMSE at pixel level | 121 |
| 6.36 | Comparison of RMSE at pixel and superpixel level | 122 |
| A.1 | Voisinage isotropes niveau superpixel | III |
| A.2 | Voisinages anisotropes niveau superpixel, résumé | V |
| B.1 | Details about stick kernel computation | XIII |
| C.1 | Orientation sampling | XX |
| C.2 | Path-based neighborhood | XXII |
| C.3 | RMSE at superpixel level | XXV |
| C.4 | RMSE at superpixel level | XXV |
| C.5 | Scene PSNR results | XXVI |
| C.6 | Scene SSIM results | XXVI |
| C.7 | Neighborhood quality map | XXVII |
| C.8 | Guidance map comparison | XXVII |
| C.9 | Depth error maps | XXVIII |
| C.10 | RMSE pixel vs superpixel level | XXX |

List of Tables

| | | |
|-----|---|------|
| 4.1 | Energy decomposition | 49 |
| 4.2 | Table of parameters | 67 |
| 6.1 | Neighborhood Confusion Matrix | 91 |
| 6.2 | Global Performance Table 1 | 123 |
| 6.3 | Global Performance Table 2 | 124 |
| 6.4 | Global Performance Table 3 | 125 |
| 6.5 | Global Performance Table 4 | 126 |
| C.1 | Mean running times with superpixels | XXIX |
| C.2 | Qualitative results obtained for two scenes | XXIX |

Chapter 1

Introduction

The popularization of image acquisition devices, together with technological advances, has led in recent years to an explosion of frameworks for the use of digital imaging data, in many fields: Graphic computing, robotics, astronomy or medical research take advantage of images collected from acquisition systems, such as cameras, telescopes, microscopes, scanners or radars for example. Computer vision is the scientific field aimed at recovering high level semantic information from these data, with the help of the increasing processing power of computers. This field includes different tasks such as methods for acquiring, processing, analyzing and understanding the digital sources.

Image acquisition gathers all techniques aimed at producing the data from digital image sensors. It ranges from the use of cameras to acquire images, videos, to the use of unmanned aerial vehicles, telescopes, radars, and from getting images or videos in the visible light spectrum to getting them in invisible wavelengths such as x rays, gamma rays, infrared, micro-waves, radio-waves. Sound waves can also be used for acquiring images, for instance with the use of sonars or ultrasonography. Each distinct channel of acquisition, that may be converted digitally into an image of scalar values, represents one specific feature of the scene. Because of the human eye sensitivity, a color image is generally encoded as a two dimension image with three channels. When a system deals with a higher number of channels, the acquired images are usually called multi-spectral or hyperspectral depending of the order of magnitude of the number of channels. In summary, image acquisition usually refers to a representation of a scene projected on a plane (the image plane), i.e. with two spatial dimensions, and at least one channel (data dimensionality). However, a scene can be recorded and/or represented using stacks of images (as defined previously), with the stack dimension representing for instance different times, different depths (third spatial axis) or different values for a given acquisition parameter.

Image (pre-)processing includes all the applications dedicated to the processing and modification of the data itself, to correct it, or enhance its quality for post analysis by humans or computer. For instance, contrast enhancement and noise removal algorithm can be used for better visual exploitation of medical images, and in-painting may help for reducing occlusions in stereoscopic imaging. Shadow compensation algorithms or color homogenization algorithms may also improve the behavior of shape from motion algorithms, where multiple photography of a building are used together for reconstructing a building in three dimensions. Many popular software solutions incorporate lots of built-in simplified image processing algorithms, which makes them available for any purpose, may it be scientific or artistic. These techniques, when used in order to improve posterior image analysis are also considered as pre-processing.

This thesis focuses on image analysis, that deals with the decomposition and recognition of some elements in the image. More specifically one of its tasks, image segmentation, deals with the problem of decomposing an image into meaningful regions whose semantic content can be used for higher level algorithms, like object classification, detection or tracking. This problem is a complex and challenging task. Depending on the application, the segmentation may typically be computed based on specific properties of the image at pixel level, for instance radiometry. However,

despite advanced image acquisition techniques and preprocessing algorithms, low-level information delivered by a single pixel is limited and prone to noise, corrupted data and all kinds of optic phenomena altering the original image. Therefore, taking into account a statistical relationship between spatially close pixels has been introduced relatively early in image processing [Geman and Geman \[1984\]](#). A classical way to handle this is to model the 2-dimensional (2D) field of pixels as a *Markov Random Field* (MRF). This allows for introducing a prior on the expected solution. Similarly, the belonging of a pixel to a region can be statistically modeled depending on the observed pixel properties. Combining these two distinct priors allows for the formulation of the ill-posed image segmentation problem as a better problem whose solution can be derived by maximizing the posterior probability of the segmentation knowing the observed image.

Variational approaches are particularly used to provide solutions, by combining the prior and conditional probabilistic models into a single parametric functional. The functional is composed of multiple energy terms embedding the priors, and must be minimized in order to produce an optimal solution. This is a challenging task: the number of dimensions of the space where the functional has to be minimized is proportional to the number of pixels of the considered image, which would make any brute-force approach irrelevant. A lot of work has been dedicated to such a minimization, considering different contexts, with different constraints and energies, under continuous or discrete formulations. The study [Szeliski et al. \[2008\]](#) gives an insight by comparing several minimization algorithms (including graph cuts) on typical vision problems, including image segmentation.

Variational approaches usually allow for some reduction of noise and image acquisition artifacts, yielding a segmentation that is closer to the expected output. However, it is well known that standard *Total Variation* (TVA) regularization (e.g. in image reconstruction [Ribal et al. \[2018\]](#)) or Potts regularization (e.g. in image segmentation [Boykov and Jolly \[2001\]](#)) behave poorly on thin structures. Detecting thin structures is very challenging because of their spatial sparsity, their small size and their potential complex geometry. Since these structures essentially consist of discontinuities, they are highly penalized. Standard TVA and Potts regularization tend to remove them first (as the weight of regularization progressively increases) and are thus not adapted to handle correctly these structures. Thin structures are ubiquitous in a number of applications (such as medical imaging) and detecting them as accurately as possible is therefore of great interest.

In parallel, due to technological advances in image acquisition, the amount and the diversity of data to exploit have greatly increased in the last years. The development of fast algorithms for exploiting this amount of data in a reasonable time is therefore of critical importance.

To reduce the computational burden, superpixel decomposition methods have been developed for grouping pixels sharing similar radiometric intensities into homogeneous regions. As they tend to preserve the image contours, superpixels are able to drastically reduce the number of elements to process while keeping the geometrical information that is lost with multi-resolution approaches. The problem of creating spatially consistent and homogeneous regions involving a tradeoff between efficiency, regularity, radiometry, adherence to boundaries and a lot of different parameters, several different algorithms for constructing them have been proposed, each of them being driven by different criteria as stated by [Stutz et al. \[2018\]](#). Due to the large number of types of superpixels, it is then necessary to integrate different metrics to further analyze their properties, from shape and size regularity, to geometric and topological constraints and measures of the semantic consistency of the resulting image of superpixels.

A major drawback of a superpixel segmentation is that the usual hypothesis of a regular topological grid structure is lost, as well as the regularity in size and shape of every element. As a result, image segmentation approaches taking advantage of superpixels must cope with these problems and introduce new methods and spatial relationships. Most often, superpixels are considered as neighbors when sharing a common border ([Cui et al. \[2018\]](#); [Fulkerson et al. \[2009\]](#); [Liu et al. \[2016\]](#); [Stawiaski and Decenci re \[2011\]](#)). [Stawiaski and Decenci re \[2011\]](#) propose to minimize an energy via graph cuts based on the watershed superpixel adjacency graph, where edges connecting two regions are weighted upon the common border length between these regions. Similarly, [Cui](#)

et al. [2018] propose to ease the classification of the high-dimensional noisy hyperspectral images by building a weighted graph based on superpixels. Pei et al. [2014] compute saliency from MRF using the same concept of adjacency, and take into account in their algorithm the second-order neighborhood to ease the propagation of information between superpixels. Some approaches make growing and merging the regions from an initial set of superpixels that they call an over-segmentation of the initial image Arbeláez et al. [2011]; Gould et al. [2009]. Other superpixel approaches use patches to analyze the spatial content over a neighboring window and find the nearest matches in a set of reference patches Giraud et al. [2017a]. In the deep learning community, Yu et al. [2015] train a deep Hough forest from a set of superpixel patches in order to detect objects in aerial images. However, both patch match and deep learning approaches require a large labeled dataset.

Additionally, some other superpixel-based segmentation approaches rely on neighborhoods that are mainly isotropic, which has been proved, at pixel level, to play an active role in the alteration of thin structures produced by regularization process Favaro [2010].

In this thesis, we focus on the regularization of segmentation obtained with anisotropic neighborhoods, based on thin structures detection on irregular lattices. This thesis contains 7 chapters, including this introduction and a conclusion.

First, we introduce the concepts used in this thesis and the formulation of the problem in Chapter 2. This includes a presentation of inverse problems in science, as well as a description of the concept of regularization. We also disambiguate the term *neighborhood*, that can have multiple significations depending on the domain. Then, we define thin structures and introduce a few examples of situations in which they are encountered. Finally, we introduce a few notations that allow us to present the global formulation of the problem as a MRF and what it entails in terms of energies.

In Chapter 3, we detail the implementation of the neighborhoods. In a first section, we present the isotropic neighborhoods at site level, from which our anisotropic neighborhoods are derived. We then list three options for estimating the presence of thin structures, and their orientations, that are: An energy-based approach, Tensor Voting (Medioni et al. [2000]), and RORPO (Merveille et al. [2018]). From each of these options, we may construct anisotropic neighborhoods, that are the main contribution of this PhD. We propose four anisotropic neighborhood constructions: one based on shapes, and its declination as a dictionary, and two based on paths, namely the target-based and path-based neighborhoods.

Then, Chapter 4 is dedicated to the implementation and numerical resolution of the problem. We detail how the energy presented in the problem formulation is minimized, summarize the parameters involved in such minimization and the tuning of these parameters. Specifically, a first section presents the use of graph cuts and the max-flow algorithms that allow us to minimize the energies of both binary and multi-label problems. Then, we present the algorithm that we can use for minimizing our functional, and a last section specifies the superpixels we use in practice, as well as a few implementation details and the parameter settings.

In Chapter 5, we apply our approach to the detection of thin structures, through a toy example based in a simulated image and an image of cracks. This allows us to compare two of the estimations of thin structures we propose, and to exhibit a few of their limits.

Chapter 6 presents the application to *Shape From Focus* (SFF). Firstly, we present the context, and secondly we specify the way we adapt our model to the data used in SFF, that are in a space in three dimensions. After describing a few details of implementation, we discuss the results and benefits of our approach in a comparison between isotropic and anisotropic regularization.

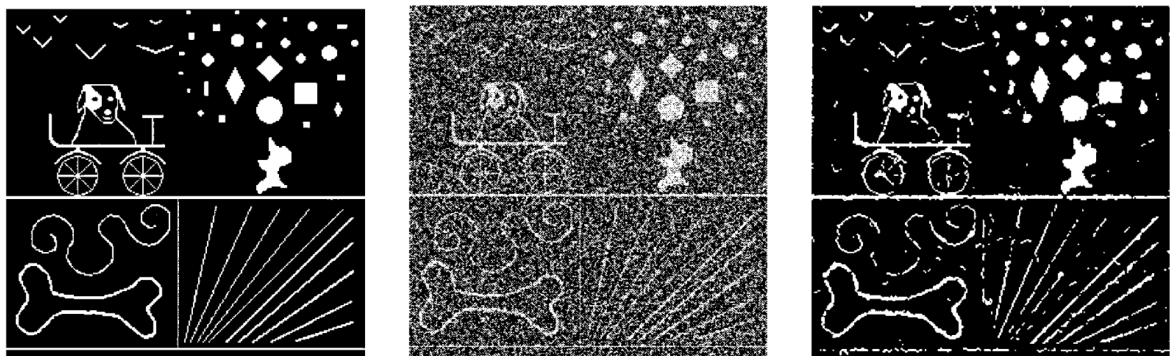
Finally we conclude this document and present some perspectives in Chapter 7.

Chapter 2

Mathematical modeling

Contents

| | |
|---|-----------|
| 2.1 Background | 6 |
| 2.1.1 Regularization and inverse problems | 6 |
| 2.1.2 The notion of neighborhood | 7 |
| 2.1.3 Superpixel segmentation | 9 |
| 2.2 Thin structures | 11 |
| 2.2.1 Definition | 11 |
| 2.2.2 Thin structure detection | 11 |
| 2.2.3 Shape From Focus (SFF) | 12 |
| 2.3 Problem formulation | 14 |
| 2.3.1 Markov Random Fields (MRF) | 14 |
| 2.3.2 Energy functional | 16 |
| Data fidelity term | 16 |
| Smoothness energy | 16 |
| 2.3.3 2D to 3D modeling | 17 |
| 2.3.4 Proposed approach | 18 |



Using the Chien-model, courtesy of [Descombes et al. \[1998\]](#).

2.1 Background

2.1.1 Regularization and inverse problems

In science, there are two kinds of problems category that can be distinguished: *direct problems* and *inverse problems*. Direct problems deal with the computation of the observable effects of a physical system. This is realized by mathematical models that allows us to predict some consequences given their causes. In electrical engineering, solving a direct problem is, for example, knowing the internal characteristics of a battery, deriving the voltage and the current that will flow if connected to a lamp or a car. The voltage and current are directly measurable physical consequences.

Conversely, the causes, that describe the state of the physical systems, are often hidden. It is a nonsense to *measure a battery* in the sense of “measuring directly its state of charge, level of wear, internal temperature, resistance”, and so on. Instead, electrical engineers may put the battery under series of tests, or measure current and voltage at the battery terminals under harmonic stress. Thanks to a good pre-established knowledge of the direct problem that models the frequency response of a battery with respect to all its parameters, one can compute the data recorded and assess the internal state of the battery. This is an example of inverse problem, that is called impedance spectroscopy. In general in science, solving inverse problems therefore allows for *characterizing* objects or physical systems.

Sometimes, multiple causes can produce the same measurable results, or said differently, a set of physical measures does not allow for characterizing in an unambiguous way the state of the physical system. It may be due to an incomplete set of measures, or to a partial knowledge of the direct problem, or to the noise that may alter the measures. Sometimes, on the opposite, trying to solve an inverse problem from a set of measures does not yield any solution: The cause may be the presence of an erroneous hypothesis in the modeling of the direct problem that over-constraint the solution, or the absence of hypothesis about noise that would allow for considering measure imprecision. In both cases, a problem that has no solution or multiple solutions is called an *ill-posed* problem. In order to compute a solution, one needs adding prior hypotheses or re-considering previously used ones to reformulate the problem as a *well-posed* problem.

For instance, let us consider the example of hyperspectral spectroscopy [Lachaize et al. \[2018\]](#). The principle of hyperspectral spectroscopy is to determine the nature of the materials imaged using a large number of spectral channels. Indeed, materials cannot be characterized only by their color in the visible spectrum, that results of pigments in the material itself. This is an under-constrained problem. The image acquisition is therefore realized in a larger set of additional wavelengths, for instance within infra-red or ultraviolet spectral domains, for which the direct model predicts different measurable responses for the different materials. In this way, the problem could be considered as well-posed, and each pixel of the image may be classified in one class or another depending on its spectral response.

Considering more wavelengths added to the acquisition, the operator could however be confronted to contradictory data, and without revising the physical model, the problem would become over-constrained. In practice, for digital imaging and especially with hyperspectral images, noise often alters the image and must be taken into account. Including noise in the analysis and establishing models for combining the apparently non consistent sources allow for computing an *optimal* solution. In this case, the optimal solution is a compromise between the different measurements, knowing the degree of reliability of each source.

The more complex the physical model is, the more difficult its analytical resolution is. Instead, the solution is often approximated via numerical resolution, and more especially by solving *optimization problems*. These problems aim at finding the point at which an *objective function*, or functional, is maximal, or minimal (over its domain).

The way the points of evaluation are selected, i.e. the optimization algorithm, is an important element in the resolution of the problem. Gradient descent, for instance, is one of these methods: from an arbitrary fixed initialization point, the gradient of the objective function in every coordinate is computed, and a new point is selected, in the steepest direction. With this simple example

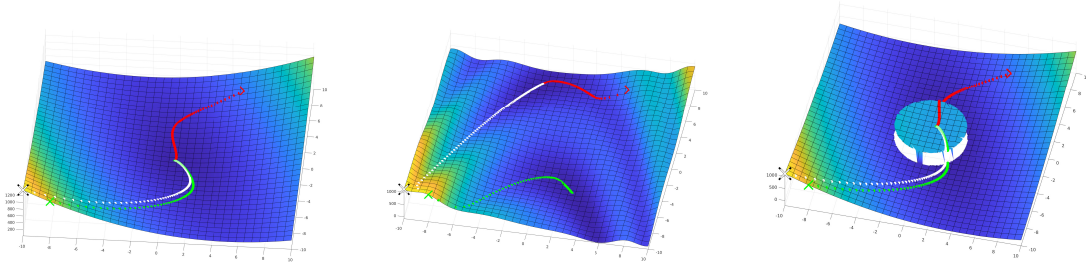


Figure 2.2 – An example of gradient descent algorithm, for a smooth and convex function (left), a function with multiple local minima (middle) and a non smooth function (right). In each image, we plot the function as a heat map on a surface, and pictured three examples of gradient descent from different initial coordinates in red, white and green. The convergence to a local minimum is possible if the function is continuous and derivable, but some limitations are already visible in this example in two dimensions: This simple algorithm may not converge, may be stuck in local minimum, and may be sensitive to initialization and to the tuning of the step size.

depicted in Figure 2.2, we can see that one of the limits of this algorithm is that it may converge to a local minimum instead of a global minimum. But also, the presence of irregular points such as in *ill-conditioned* problems dramatically reduces the chances of the algorithm to converge. This is the reason why the presence of spikes, irregular points and discontinuities in the physical model is discouraged, and the smoother and the more regular is the functional, the easier it is to optimize it numerically.

Problems in the field of image analysis share a common specificity: They usually deal with billions of variables, that constitute an image. Until now, we have focused on some problems that solve the inverse problem in each pixel independently.

But the issue that is the most likely to happen is that noise tends to degrade the image and alter the measures, therefore preventing correct solving of the inverse problem. We picture this with a simple toy example of *local* (or *blind*) segmentation in Figure 2.3. The image in the middle presents isolated pixels in its labeling that we intuitively discriminate as erroneous.

Usually, a scene is not composed of one-pixel sized physical objects, juxtaposed without interactions or correlation between each other. Instead, an image usually represents a finite set of objects, each of them composed of multiple pixels. These objects display either homogeneous properties like color or distance to the camera, or textured but somewhat regular geometric structures. This knowledge leads to an additional hypothesis on the image that is often called *smoothness prior*: There is statistically more chances that two adjacent pixels belong to the same object, because objects are statistically larger than two pixels, and tend to present homogeneous physical properties.

The problem formulated with this new prior is called a *regularization* problem: Instead of solving each pixel as a separate sub-problem, the prior introduces a model of local configurations on the surroundings of the pixel, and isolated pixels (presenting differences with respect to their neighbors) are more likely to be considered as errors and regularized, as illustrated by the third image of Figure 2.3.

Having introduced the concepts of numerical resolution, optimization, and regularization, let us now define more precisely the notion of neighborhood and the framework used in this thesis.

2.1.2 The notion of neighborhood

The term of neighborhood is a central concept that requires some discussion. We encounter the notion of neighborhood and similar aspects in several situations both in science in general and in this thesis.

In everyday life, a neighborhood is a geographically localized community within a larger city, more specifically defined by sociologists as “a specific geographic area and functionally as a set of

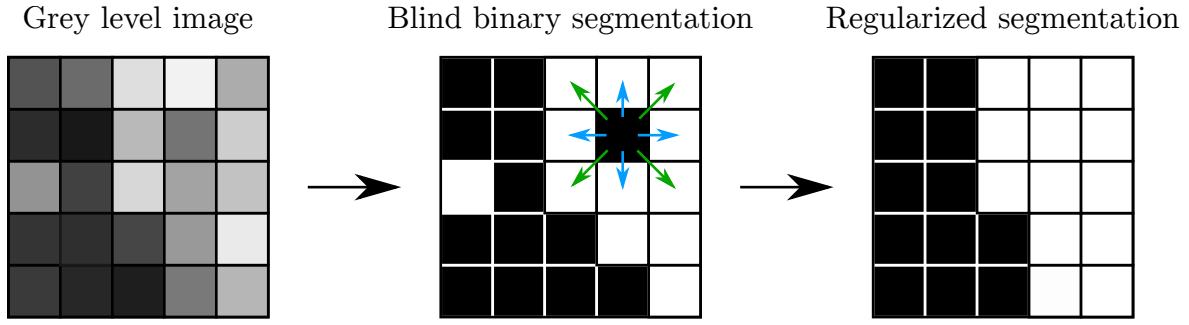


Figure 2.3 – For a toy example image (left), we present the result of a segmentation without regularization (or blind, in the middle), and of segmentation with regularization (right). With the smoothness prior, the problem is better-posed: Regularization allows for improving the labeling of some of the erroneous pixels, by formulating a prior of homogeneity of the labeling in their neighborhood. For instance in the blind segmentation image, the 4-adjacency or 8-adjacency neighbors of the isolated black pixel are indicated by the blue arrows and by the union of blue and green arrows, respectively, and allow to regularize this pixel as a white pixel.

social networks. Neighborhoods, then, are the spatial units in which face-to-face social interactions occur” (Schuck and Rosenbaum [2000]). This definition is not completely unrelated to our topic. In the same way that the social neighborhood depends on the notion of “face-to-face” social interactions between people encouraged by spatial proximity, but also local urban planning, that determines its geographic extent, our notion of neighborhood is also a mathematical construction that puts into equation the “element-to-element” interactions, where the elements are the image pixels in our case.

In mathematics, a neighborhood is a basic concept of topological spaces. Intuitively, if something’s location is at some point, for any displacement in any direction, there exists a small distance of displacement that makes the new location to be part of the neighborhood of that point. The neighborhood in topology is then the union of all these possible destinations and the origin, and can be infinitely small, for instance with continuous spaces: In \mathbb{R} , a neighborhood of $\{1\}$ is $]1 - \epsilon, 1 + \epsilon[$ with $\epsilon \in \mathbb{R}_{>0}$. We note that with this mathematical definition, the neighborhood of a point includes the point itself.

Because our problem of image segmentation has a discrete geometry, we also need to mention the definition of a neighborhood in graph theory that is closer to our formulation. In graph theory, a graph is a set of elements named vertices, also called nodes or points, connected by multiple edges, also called links. Those edges can link vertices symmetrically, for undirected graphs, or asymmetrically, thus defining directed graphs. The definition of adjacency in graph theory is then as follows: In undirected graphs, two nodes connected by an edge are said to be adjacent, and in directed graphs, a vertex t is adjacent to a vertex s if there is a directed edge from s to t . Here, the first vertex of the edge s is called a *predecessor* of the second vertex t and t is a *successor* of s . From this notion, the neighborhood of any vertex s is defined as the union of its successors (Heijmans et al. [2005]).

In image processing, the elements that are processed are usually pixels, and as introduced a bit earlier with the notion of smoothness prior, regularizing the solution corresponds to taking into account the spatial configurations of the elements over a neighborhood, that is often derived from a stationary adjacency relationship such as 4-connectivity for instance, as depicted in Figure 2.4. With 4-connectivity, each pixel is connected to the four pixels directly above, below, at its left and at its right. The 8-connectivity also involves the pixels in the diagonals. Because the problem can thus be seen as a graph, the neighbors of a pixel can be derived from the graph theory definition as, for each vertex s , the union of the successors of s . We note that depending on the reference, some authors may indifferently mention such neighborhoods as *4-neighborhood* (Goldfarb and Yin [2009]), *4-connected* or *4-adjacency* (Lézoray and Grady [2017]). We will use the latter denomination in this paper.

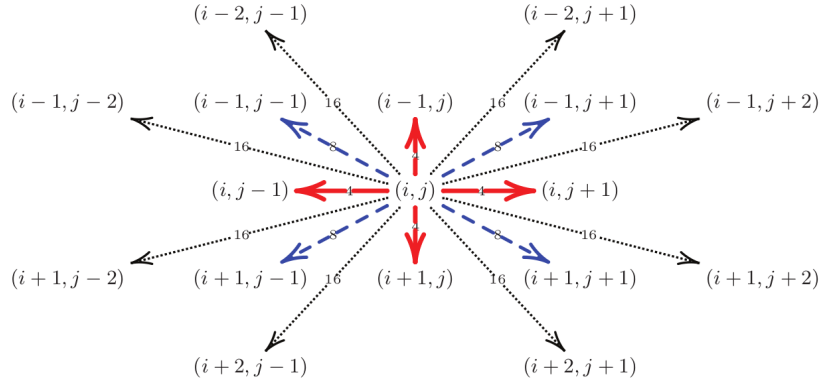


Figure 2.4 – This graph, courtesy of Goldfarb and Yin [2009], represents the neighbors of a pixel (i, j) by connecting them with arrows. These neighborhoods are commonly used in image processing: thick red arrows represent 4-adjacency neighborhoods, while sites connected with blue dashed lines are added to the 8-adjacency neighborhoods. The 16-adjacency neighborhood gathers all the sites of this example. We note that with Goldfarb and Yin [2009] formulation, each type of arrow has a distinct weight in the neighborhood.

In our work, we aim at remaining generic with respect to the use of pixels or superpixels, therefore we consider the term of *sites*. Because we introduce several notions more or less related to the concept of neighborhood in its broader acceptance, we now explicit the usage of this term.

The neighborhood definition provided by graph theory is the one we use for the term *neighborhood* in this thesis. We encounter these neighborhoods in our optimization problem, and discuss the way they are constructed. However, additional neighborhood-like notions are useful.

Another type of neighborhood-like relationship is encountered in the implementation of our neighborhood construction. It relies on spatial *adjacency* as an indicator of whether the sites share a common border or not. When working at pixel level, this notion reduces to 4-adjacency or 8-adjacency, see Figure 2.4. If the sites are superpixels, we decide to use 4-adjacency at pixel level for defining their borders and find the adjacent superpixels from adjacent pixels that belong to two different superpixels. Two superpixels having this common border are said to be *adjacent*.

Later in this manuscript, we encounter a similar notion for path based morphological operators. We will refer to this case as *connectedness*, since the definition depends on a function that *connects* the elements, see Chapter 3.2.3.

With these three distinct notions, we aim at defining new anisotropic neighborhoods for performing anisotropic regularization on sites that may either be pixels or superpixels.

2.1.3 Superpixel segmentation

Initially introduced by Ren and Malik [2003], the term *superpixel* refers to groups of pixels that share the same low-level properties, like for instance being similar in color, or sharing close locations. Existing before the definition of the term, those superpixels have been used in a wide range of applications as they answer two common issues in image processing. Firstly, superpixels allow for reducing the computational complexity of algorithms, when compared to the cost of treating numerous pixels. Secondly, this pre-processing yields regions that are consistent with the content of the image, and therefore allows for the extraction of higher-level features than when working at pixel-level.

Various algorithms define different kinds of superpixels that embed different properties, such as the adherence to the boundaries of the objects, the compactness or convexity of the resulting superpixels, their regularity, or the smoothness of their boundaries. More generally, when using superpixels, the authors agree that they should meet the requirements listed in Achanta and Susstrunk [2017]; Stutz et al. [2018]:

- **Partition:** Superpixels define a partition of the set of pixels;
- **Connectivity:** Superpixels are connected sets of pixels;

- **Boundary adherence:** Superpixels preserve the image boundaries;
- **Compactness, Regularity, Smoothness:** In the absence of object's boundaries, the superpixels are expected to be compact, placed regularly and to exhibit smooth boundaries;
- **Efficiency:** Superpixels should be generated efficiently in terms of memory and computation time;
- **Number of superpixels:** The number of generated superpixels should be controllable.

For each algorithm, various low-level features are extracted from the original image to ensure that these properties are, by construction, homogeneous within the derived superpixels. These features may include the value of each pixel in a selected colorspace (Grayscale, RGB, CieLAB, YCrCb, *etc.*), the distance (Euclidean, geodesic, *etc.*) from a given set of seeds, the belonging to a boundary map (or alike), the derivatives of the original image, or any other feature that the author finds relevant. Some algorithms involve parameters that allow the user to balance between different behaviors of the algorithm, thus impacting the superpixel decomposition. Indeed, there is often a trade-off between the different properties that the superpixels are expected to fulfill.

It is possible to define some criteria that allow us to compare quantitatively the performance of the superpixel algorithms with respect to each property. A lot of different metrics coexist, some of them being shared by multiple authors, sometimes with slightly different definitions. While a few of them are computed only based on the image and its segmentation, a majority of them also require a ground truth. Most of these criteria strongly vary with the number of generated superpixels. Some benchmarks [Stutz et al. \[2018\]](#); [Wang et al. \[2017\]](#) thus vary the number of superpixels, and compute the area under curve for more accurate information about the fidelity of the generated superpixels with respect to each of the expected properties.

In practice in our works, after a few comparisons, we focus on the superpixels provided by an algorithm called [Extended Topology Preserving Segmentation \(ETPS\)](#) ([Yao et al. \[2015\]](#)), since it is energy based (as the general framework adopted for our work) and offered relatively smooth and regular superpixels.

Using superpixels in segmentation and in image processing introduces a lot of challenges, due to the loss of regularity of the lattice formerly composed by the pixels of the image. Usually, image processing algorithms, such as Gaussian blurring, morphological openings or closing, mean filtering, path openings, Laplacian filtering and many others make an intensive use of the topological regular structure of the pixel grid. In the case of mathematical morphology, [Asplund et al. \[2019\]](#) have addressed the problem of mathematical morphology on irregularly sampled data in one way, by duplicating the sampling points to correctly spread grayscale data to the borders of the support of the structuring function. However, this approach leads to an increase of the number of samples without solving the problem of the definition of neighborhoods on irregular lattices.

In this thesis, we address this problem in multiple ways. Firstly, we propose various new neighborhood constructions that are compatible with both pixels and superpixels, which is one of the core contribution of this thesis. Besides, we modify some of the existing algorithms to use them on superpixels, mainly by using the averaged value of the physical properties of the pixels inside a superpixel. This allows us to still make use of these algorithms without computing them at pixel level. For example, the position, or the color, of a superpixel are approximated as the barycenter of the positions, or colors, of the pixels included. We also aim at providing a generic formulation for both pixels and superpixels, therefore we detail these specific behaviors throughout the manuscript of this thesis by mentioning them as sites.

One important fact about using superpixels is that it increases the size of the elements of the image, therefore the objects' size in terms of number of elements decreases. Since we focus our interest on anisotropic segmentation, we now introduce thin structures and some examples of applications that involve them.



Figure 2.5 – Example of thin structures in 2D. The first image (left) is a retinal image extracted from the DRIVE Dataset⁰: The thin structures in this image are the blood vessels. The middle image is a fingerprint, and the right image features a deer, courtesy of Tran et al. [2019]. While a deer itself may not be thin, its antlers typically are and may be challenging to segment.

2.2 Thin structures

2.2.1 Definition

A thin structure has very small size in at least one dimension compared to the other one(s).

We present some thin structure examples in two dimensional images in Figure 2.5. In this context, thin structures are curves, that extend in one dimension. In 3D, thin structures may also be planes, as long as their thickness can be considered as very small compared to one of their other dimensions. A common example of thin structures is blood vessels, that are widely considered in biomedical imaging.

In this thesis, we focus on the two following applications: Crack segmentation and Shape From Focus (SFF). The aim of crack segmentation is to specifically detect thin structures (the cracks) in textured backgrounds, through a binary segmentation. Thin structures may or may not be involved in SFF, depending on the scene that is captured. However, we picked this application since it challenges our anisotropic analysis with the introduction of a third dimension and a larger set of ordered labels.

2.2.2 Thin structure detection

We illustrate thin structure detection in the context of crack detection. Crack detection is a particularly useful application for maintenance and safety in general, since fixing a crack before its deterioration reduces a lot the costs of reparation and reduces the occurrence of unexpected accidents. It is the case in particular with roads and highways, where for instance the cumulative length of the French road network is around one million kilometers. Since image-based technology can provide a non invasive, safe, efficient and economical way for performing road crack detection, various approaches have been developed for this task.

A crack basically takes the appearance of a discontinuity in the background corresponding to the material (asphalt, concrete, etc.). The intensity in the crack is usually lower than that of the background, which naturally tends to favor the use of thresholding methods for the detection of cracks. However, the materials often exhibit highly textured surfaces, that drastically affect and degrade the performance of thresholding methods that consequently only produce disjoint crack fragments. A naive approach would cope with that phenomenon by performing noise removal algorithms, but by nature, a crack is a thin structure: Its size is very small orthogonally to its propagation direction, and it can therefore easily be concealed by noise or hidden in the texture. For

⁰The DRIVE Dataset is currently available at: <http://www.isi.uu.nl/Research/Databases/DRIVE/>

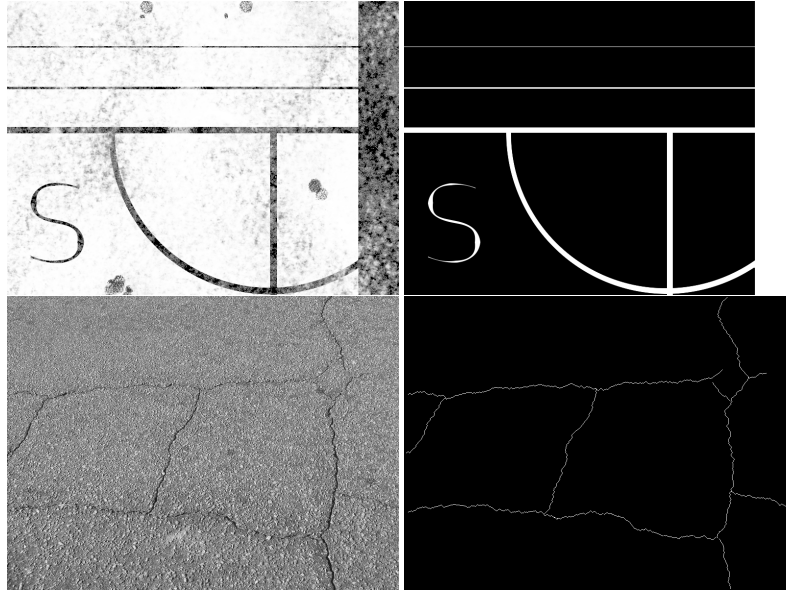


Figure 2.6 – Two examples of image used for crack detection on the left hand side, and the associated ground truth on the right hand side. The first row is a simulated image, and the second row is a crack image from the CrackTree dataset (Zou et al. [2012]). The crack that we expect to detect is the thin dark line snaking in the image, whose curvature makes it difficult to detect.

these reasons, crack detection is a problem that is difficult to apprehend, and we therefore propose to challenge our anisotropic regularization model on such task.

Crack segmentation process can be summarized as follows. First, a device realizes an acquisition of an image (in two dimensions) of the material to analyze. Then some pre-processing steps may be applied on the image, for helping a better detection of the cracks: Blurring, mean filtering or opening for instance. In the idea of presegmenting the problem in superpixels, the crack image is used for generating the superpixel image, sometimes with additional pre-processing aiming at smoothing the resulting superpixels. We expect the segmentation algorithm to finally discriminate accurately the sites that contains parts of the crack by labeling each site with “crack” or “non crack”. The ground truth is a binary image in which pixels are labeled the same way. However, sites may be larger than one pixel and detection algorithm may be allowed to have slight misalignment, therefore we measure the quality of the produced labels with a small tolerance to position errors.

Our data comes from the CrackTree dataset Zou et al. [2012], to which we have added a few simulated images, as shown in Figure 2.6. The simulated images contain simple shapes, forming the ground truth, to which are manually added textured noises and some imperfections to simulate acquisition noise. The data of the CrackTree dataset are some images of crack in the concrete, with variable angle of exposition, variable textures and perturbations such as shadows. In addition, the shooting angle induce anisotropic and non-stationary properties in the images: Some vertical cracks are more visible due to the alignment of their 2D plane of extension with the optical axis, while some horizontal cracks disappear by obfuscation of the depths of the crack, and the texture of the concrete is bigger in the foreground, due to the non alignment of the image plane with the surface captured. The ground truth is a manually labeled image in which all the pixels labeled “crack” point to the center of the crack along its small dimension. Given that a crack may be sometimes more than one pixel large, and sometimes less than one pixel large, the ground truth is by nature a subjective choice made by the human observers.

2.2.3 Shape From Focus (SFF)

SFF is an optical passive method for extracting the 3D shape of a scene from multiple images captured with varying focus settings (Nayar and Nakagawa [1994]), using a single camera. For doing so, SFF benefits from prior knowledge of optic phenomena happening during image acquisition,

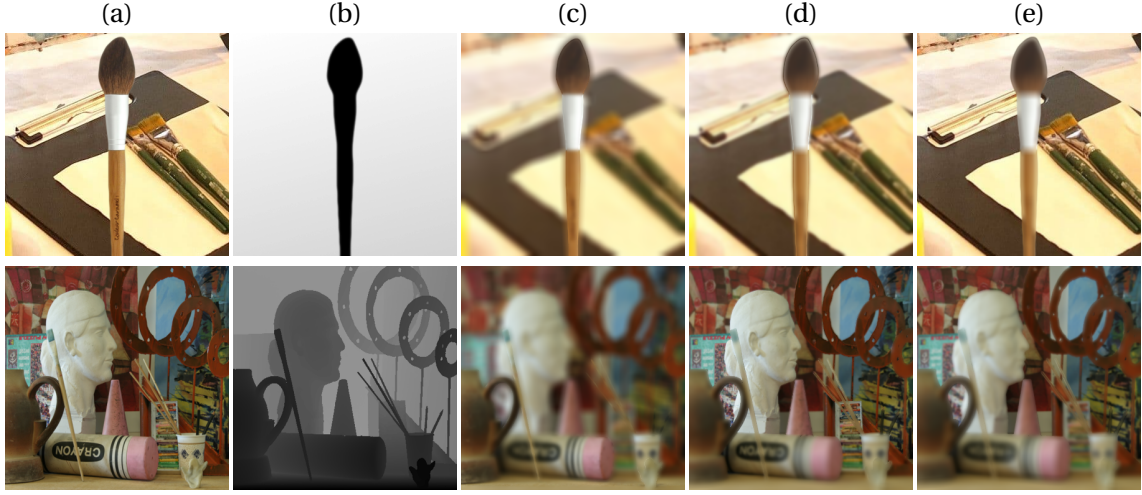


Figure 2.7 – The data used for our Shape From Focus application. The all-in-focus image (a) and the ground truth (b) allow us to simulate defocused images (c,d,e) from which we try to recover the ground truth. The first row shows our toy example *Brush1*, and the second one is *Laundry1* from the Middlebury dataset.

the main one being: The more an object is close to the object focal plane of the optical system (i.e., the more it is *focused*), the more it appears sharp. Conversely, the more an object is far from this object focal plane (i.e., *defocused*), the more it appears blurry. By definition, because the depth of each point of the scene is computed from a set of images that are an experimental observation of the blurring phenomenon, SFF is an ill-posed inverse problem. It is applicable in many applications including industrial inspection, micro manufacturing, robotic control, 3D model reconstruction, medical imaging systems and microscopy, and we will detail its theoretical background and processing steps in the Chapter 6.

In brief, a sharpness operator [Pertuz et al., 2013] is used to measure the sharpness in each point of the volume. For each 2D coordinate, selecting the slice image giving the maximum of this measure allows us to produce an initial depth map. At this point the depth estimation is of limited quality and authors (Gaganov and Ignatenko [2009]; Mahmood and Lee [2020]; Moeller et al. [2015]; Nair and Stewart [1992]; Nayar and Nakagawa [1994]; Ribal et al. [2018]) aim at improving this depth map by working on several pre- or post-processing.

In our work, we focus on anisotropic regularization with non regular image lattice, thus placing this application in the combinatorial optimization framework.

The dataset on which we focused for our experiments is derived from the Middlebury college one from 2005 and 2006 (Scharstein et al. [2001]). This dataset provides, for various realistic scenes, accurate depth maps as well as colored all-in-focus images, with several available exposures and illumination settings. Among them, we have selected the intermediate exposure, the lowest illumination and smallest image resolution, for two views denoted 1 and 5. The unknown depth values due to occlusions have been estimated by the median value of the surrounding depths. We selected some scenes, to have an overview of the performance of our approach on most of the situations available in the dataset.

We generate for each scene a sequence of 50 images with varying focus, using the code provided by Pertuz et al. [2013]¹ run with default parameter values: Each pixel of the all-in-focus image is blurred depending on the distance between its actual depth and the image focal plane.

An example of all-in-focus images from the dataset as well as a toy example and their simulated defocused images are presented in Figure 2.7.

¹Defocus simulation algorithm currently available as a Matlab source on MathWorks file exchange at <https://fr.mathworks.com/matlabcentral/fileexchange/55095-defocus-simulation>.

2.3 Problem formulation

To sum up the specific aspects introduced in the first sections of this chapter, we place ourselves in the domain of combinatorial optimization, and aim at improving the regularization of the segmentation of challenging scenes in image processing involving thin structures and irregular lattices. This leads us to take into account, for each site, the local configuration of nearby sites, which is usually done with the help of MRF, and results in the introduction of the notion of anisotropic neighborhoods.

2.3.1 Markov Random Fields (MRF)

The configurations of the sites $\{s\}_{s \in \mathcal{S}}$ of an image define a random field that may be described in two different but equivalent ways, as formulated by Spitzer [1971]. The first approach originates from statistical mechanics and defines the random field as a Gibbs field. Such formulation describes each site as a particle in interaction with other sites, the interaction being described by a potential. The second approach is the one of MRF, and is a purely probabilistic definition based on a set of properties about conditional independence of the variables of the field given their neighbors.

MRF in image processing analyzes the image as a graph composed of a set of elements \mathcal{S} , that we call sites, connected to their neighbors by edges. The sites $s \in \mathcal{S}$ may either be pixels or superpixels, and constitute in both cases a partition of the image. If we denote by \mathcal{P} the set of pixels of the image, we note that $\mathcal{S} \subset 2^{\mathcal{P}}$, where the notation 2^X is the powerset of X , and the partition condition can be written $\cap_{s \in \mathcal{S}} s = \emptyset$ and $\cup_{s \in \mathcal{S}} s = \mathcal{P}$. We note that the set of pixels can be described as $\mathcal{P} \subset \mathbb{R}^N$ where N is the dimension of the geometrical space, for which an orthonormal reference frame is the set of vectors $(\mathbf{e}_0, \dots, \mathbf{e}_{N-1})$. In most of our application, $N = 2$ (2D) or $N = 3$ (3D).

The concept of MRF derives from a specific model, the Ising model, in 1925. Ernst Ising illustrates his model on the example of a set of atoms of ferromagnetic materials he was working on, where each element could either be in the position of a “spin up” or “spin down”. The spin of each site is the hidden and random variable, and we denote the field by $\mathbf{u} = \{u_s\}_{s \in \mathcal{S}} \in \mathcal{C}^{\mathcal{S}}$, where \mathcal{C} denotes the set of states each site can take.

In the experimental observations, the spins would naturally tend to align (attractive case) or to alternate (repulsive case), while still being subject to the effects of external magnetic fields. The idea was however to formulate a purely mathematical model that would describe the probability of each configuration. Since the focus is on the probability of each configuration and not the probability of the observation given a configuration, this probability corresponds to the prior on the configuration, $\mathbb{P}(\mathbf{u})$.

For modeling this prior, Ising defines the probability of each configuration from an *energy term*, under the assumption that only local interactions between neighboring spins needed to be taken into account. This hypothesis is the local property of MRF, $\forall s \in \mathcal{S}$:

$$\mathbb{P}(u_s | \{u_p\}_{p \in \mathcal{S} \setminus \{s\}}) = \mathbb{P}(u_s | \{u_p\}_{p \in V(s)}), \quad (2.1)$$

where $V(s)$ is the set of neighbors of s . This means that the knowledge of the local interactions (within neighborhood $V(s)$) is sufficient to derive the probability in a given site. We define mathematically a neighborhood V as an application that maps any site to a subset of sites (called its neighbors), and denote it $V = \{\mathcal{S} \rightarrow 2^{\mathcal{S}}\}$.

Then, the configuration probability for each site $s \in \mathcal{S}$ with labeling u_s depends on the configuration of \mathbf{u} over its neighbors $V(s)$.

To go further in the probability expression, let us introduce the set of *cliques*, that is a set of site subsets, denoted as $\mathcal{N} \subset 2^{\mathcal{S}}$. Usually, like in the Ising model, these cliques only contain pairs of adjacent sites and are therefore named 2-order cliques. In the context of ferromagnetic materials, the random field is a binary field, so that four configurations probabilities are computed (two for aligned spins, two for opposite spins) for each of the cliques including the site s .

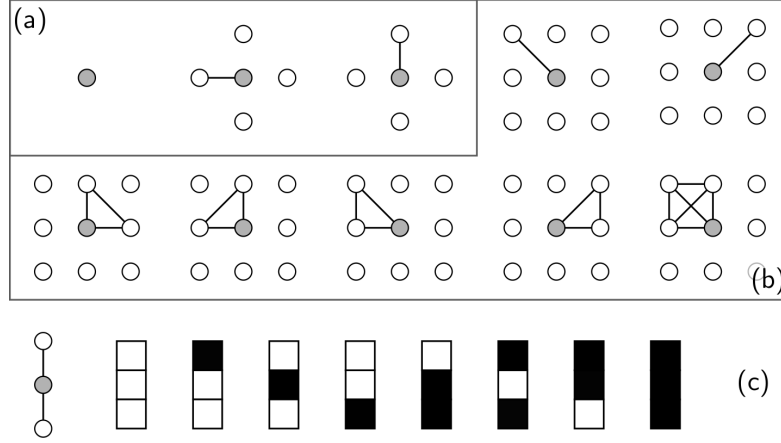


Figure 2.8 – Examples of cliques of a MRF, for a regular lattice in 4-adjacency (a) and in 8-adjacency (a and b). We note that the number of cliques available increases rapidly with the adjacency, from singletons and second order cliques with 4-adjacency to third and fourth order cliques in 8-adjacency, in the second row. In the last row, we show an example of the eight configurations of another type of 3-order cliques. These cliques, while introducing more complexity to the model, allow us to take into account more complex phenomena. For instance, the third and sixth configurations should be the less likely, while first and last configurations should be of lower potential energy.

However, in the attractive case, the ground state, or the highest probability configurations, according to the smoothness prior modeled by 2-order cliques, are uniform images. Higher order cliques have been studied for taking into account more complex spatial configurations of real images. [Geman and Geman \[1984\]](#) illustrate how the diversity of clique types increases while growing the radius of a neighborhood on a regular lattice. For instance, [Descombes et al. \[1995\]](#) propose to integrate up to 52 energy classes for the 512 configurations of 3×3 -cliques in their chien-model, that allows for discriminating phenomenon such as parallel stripes and object edges with oblique angles and giving them low energy levels. We give examples of two, three and fourth order cliques in Figure 2.8. Interesting illustrations and more thorough explanations on MRF are available in [Kindermann and Snell \[1980\]](#). In our case though, since solving problems with higher order cliques is rather complicated in terms of hyperparameter setting, we rather propose to focus on neighborhood adaptive definition while keeping modeling based only on second order cliques.

When the probabilities of all the configurations are strictly positive, it is possible to formulate them as the negative exponential of a potential energy. This boils down formulating the problem as a Gibbs field, that is equivalent to a MRF where the potential energy of a configuration is defined as the negative logarithm of its probability.

In addition to the prior probability of each configuration of \mathbf{u} modeled thanks to the MRF prior, the observation of an image $I \in \mathcal{P} \in \mathbb{F}$ that is a realization from a configuration \mathbf{u} allows us to compute the likelihood $\mathbb{P}(I|\mathbf{u})$, which also describes the modeling of the noise in the acquisition process.

With the Bayes theorem, we can compute the *a posteriori* probability as:

$$\mathbb{P}(\mathbf{u}|I) = \frac{\mathbb{P}(I|\mathbf{u}) \mathbb{P}(\mathbf{u})}{\mathbb{P}(I)} \propto \mathbb{P}(I|\mathbf{u}) \mathbb{P}(\mathbf{u}) \quad (2.2)$$

where $a \propto b$ refers to the proportionality relationship: a scalar value $y \in \mathbb{R}$ exists such that $a = y \times b$, and $\mathbb{P}(I)$ is the probability of observing the image I . Since I is a constant we only have to maximize the probability $\mathbb{P}(\mathbf{u}|I)$ with respect to \mathbf{u} up to this constant factor. This more likely configuration is the one satisfying the maximum a posteriori criterion, in the Bayesian framework chosen to formulate our problem.

2.3.2 Energy functional

In our case, we consider the maximum a posteriori estimator for a probability expressed with respect to a MRF $\mathbf{u} \in \mathcal{C}^{\mathcal{S}}$ with an anisotropic neighborhood $V \in \{\mathcal{S} \rightarrow 2^{\mathcal{S}}\} = \mathbb{V}$. Since all the configurations have a non null probability by hypothesis, this random field is also a Gibbs field, and we express the probability as the negative exponential of a potential energy. This energy is a functional $F : (\mathcal{C}^{\mathcal{S}} \times \mathbb{V}) \rightarrow \mathbb{R}$, that we want to minimize with respect to both its variables, i.e. the label field and the neighborhood local configurations. This energy is written:

$$F(\mathbf{u}, V) = E_1(\mathbf{u}) + \alpha E_2(\mathbf{u}, V), \quad (2.3)$$

where $\alpha \in \mathbb{R}_{>0}$ is a parameter controlling the balance between the smoothness prior E_2 and the data fidelity term E_1 . Note that $E_1(\mathbf{u})$ also depends on I , but that the terms that are fixed are often omitted in the functional, for clarity. This formulation is very common in the literature, and is used in most regularization problems, with several options for the terms E_1 and E_2 that depend on the application.

In the following of this section, we detail the latter ones in our context.

Data fidelity term

The data fidelity term $E_1(\mathbf{u})$ (in Equation (2.3)) is the energy term corresponding to the likelihood $\mathbb{P}(I|\mathbf{u})$ (Equation (2.2)). We note that $I(s) \in \mathbb{F}$ in the site image is the average value of $I(p) \in \mathbb{F}$, $\forall p \in s$ with $s \in \mathcal{S}$, where the feature space $\mathbb{F} = \mathbb{R}^N$ depends on the application. This feature space can be a scalar, for grayscale images, or depth images, or a three-dimensional vector, for RGB images for instance. At pixel level, very popular models adopt such a statistical approach. Some of them realize an approximation of the probability density function $\mathbb{P}(I(s)|u_s)$, $I(s) \in \mathbb{F}$ being the observation and $u_s \in \mathcal{C}$ the class of site s . This approximation can be based on the histogram of a set of pre-labeled reference pixels [Boykov and Jolly \[2001\]](#), for instance with [Parzen \[1962\]](#) kernels.

For each class $c \in \mathcal{C}$, we assume the probability density function to be a Gaussian, defined by its mean value $\mu_c \in \mathbb{F}$ and standard deviation $\sigma_c \in \mathbb{R}_{>0}$. In addition, we suppose averaged pixels of a site are statistically independent conditionally to the label field, so that $\forall s \in \mathcal{S}$:

$$\mathbb{P}(I(s)|u_s) = \left(\frac{1}{\sigma_{u_s} \sqrt{2\pi}} \exp \left(-\frac{\|I(s) - \mu_{u_s}\|_2^2}{2\sigma_{u_s}^2} \right) \right)^{A(s)},$$

where $\|\cdot\|_2^2$ is the squared L^2 norm in \mathbb{F} and $A(s)$ is the area of the site, in pixel unit.

When considering the negative log likelihood for this normal distribution, we have, for any site $s \in \mathcal{S}$:

$$-\log(\mathbb{P}(I(s)|u_s)) = \frac{A(s)}{2} \log(2\pi\sigma_{u_s}^2) + A(s) \frac{\|I(s) - \mu_{u_s}\|_2^2}{2\sigma_{u_s}^2}.$$

This provides the proposed elementary term of E_1 corresponding to one site $s \in \mathcal{S}$, up to a constant:

$$E_{1_s}(u_s) = A(s) \frac{\|I(s) - \mu_{u_s}\|_2^2}{2\sigma_{u_s}^2} + A(s) \log(\sigma_{u_s}).$$

Finally, under the hypothesis of conditional independence of each site,

$$E_1(\mathbf{u}) = \sum_{s \in \mathcal{S}} E_{1_s}(u_s).$$

Smoothness energy

Energy term $E_2(\mathbf{u}, V)$ corresponds to the smoothness prior on the labeling \mathbf{u} and its definition is based on a neighborhood, $V \in \mathbb{V}$. The idea of this neighborhood originates from the model of MRF on 2-vertex cliques, that can be oriented when the probabilistic dependencies are not reciprocal.

With the hypothesis of \mathbf{u} being a MRF with any configuration having a non null probability, \mathbf{u} follows a Gibbs distribution and therefore:

$$\mathbb{P}(\mathbf{u}, V) \propto \exp\left(-\beta \sum_{(s,t) \in \mathcal{N}} (E_{2,s,t}(u_s, u_t))\right).$$

Note that in previous equation, we assume that \mathcal{N} contains oriented cliques. If it was not the case, then a factor 1/2 would appear later. The above equation then becomes:

$$\mathbb{P}(\mathbf{u}, V) \propto \exp\left(-\beta \sum_{s \in \mathcal{S}} \sum_{t \in V(s)} E_{2,s,t}(u_s, u_t)\right).$$

The negative logarithm of this quantity is the energy $E_2(\mathbf{u}, V)$:

$$E_2(\mathbf{u}, V) = \sum_{s \in \mathcal{S}} \sum_{t \in V(s)} E_{2,s,t}(u_s, u_t). \quad (2.4)$$

When the labels are not ordered, we adopt in our experiences the Potts model Wu [1982], with a weighting function $W : \mathcal{S}^2 \mapsto \mathbb{R}$ that models the strength of interaction between neighboring sites. The definition of $E_2(\mathbf{u}, V)$ is thus the following:

$$E_2(\mathbf{u}, V) = \sum_{s \in \mathcal{S}} \sum_{t \in V(s)} W(s, t) \mathbb{1}_{\{u_s \neq u_t\}},$$

where,

$$\mathbb{1}_{\{a \neq b\}} = \begin{cases} 0 & \text{when } a = b, \\ 1 & \text{otherwise.} \end{cases}$$

With ordered labels, such as in reconstruction problems, we consider the total variation:

$$E_2(\mathbf{u}, V) = \sum_{s \in \mathcal{S}} \sum_{t \in V(s)} W(s, t) |u_s - u_t|. \quad (2.5)$$

This energy term favors piecewise constant labeling, and allows for smoother spatial evolution of the labels than Potts model. However, total variation tends to introduce staircase effect, i.e. flat regions separated by artifact boundaries.

2.3.3 2D to 3D modeling

In this thesis, since we work with 2D segmentation, we describe our problem with the formulation of 2D neighborhoods, while we could easily imagine an extrapolation to the 3D or higher dimensional cases. For some applications, we can still work with data with spatial dimensions $N > 2$, however for computational resources and also for simplicity with the notations, descriptions and illustrations of neighborhood construction, we implement 2D neighborhoods and replicate the two-dimensional superpixel partition over the other dimensions.

Therefore, while the set of sites in 2D is denoted \mathcal{S} , the set of sites in 3D is denoted \mathcal{S}_3 and can be derived from:

$$\mathcal{S}_3 = \{t / \exists s \in \mathcal{S}, h \in \llbracket 0, \Delta_h \rrbracket / t = \{p \in \mathcal{P} / \exists q \in s, \overrightarrow{qp} = h\mathbf{e}_2\}\}, \quad (2.6)$$

where Δ_h is the dimension of the third axis \mathbf{e}_2 , and the notation $\llbracket a, b \rrbracket$ refers to the set of consecutive integers, $\forall a, b \in \mathbb{Z}$, $\llbracket a, b \rrbracket = [a, b] \cap \mathbb{Z}$. With this definition, the set \mathcal{S}_3 is such that for any site $t \in \mathcal{S}_3$, there exists a site $s \in \mathcal{S}$ that is the translated set of pixels of t with translation $-h\mathbf{e}_2$.

In the next chapters, we introduce most of the modeling in 2D when the derivation to 3D is trivial, and present more details for ambiguous situations.

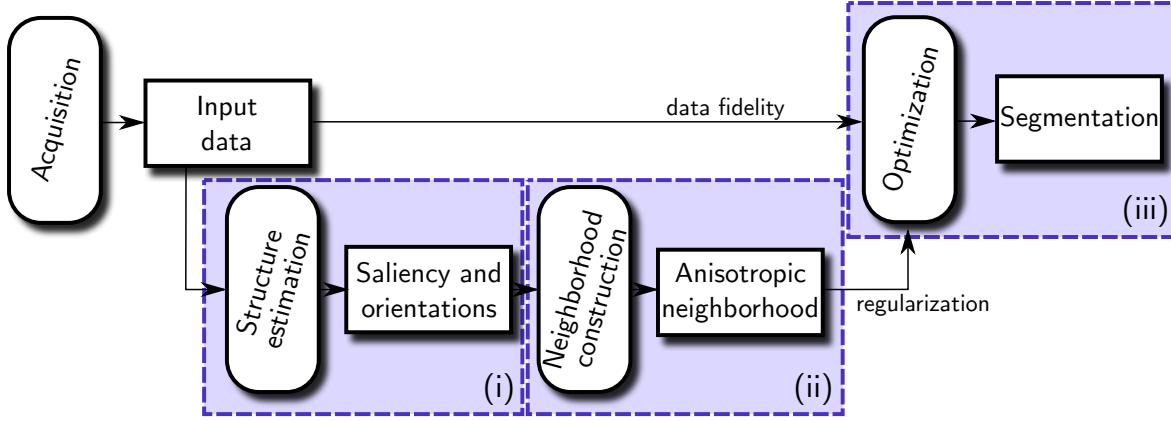


Figure 2.9 – Flowchart of the proposed approach: We propose a modular formulation where our major contribution lies in the construction of anisotropic neighborhoods, in block (ii). Block (iii) allows for minimizing the functional and thus producing an optimal segmentation, while block (i) is aimed at detecting the presence and orientations of thin structures of the image.

2.3.4 Proposed approach

Our contribution is to propose anisotropic neighborhoods suited to thin structures, in the context of the regularization of a segmentation computed on an irregular grid of sites. We present our approach as a modular approach. The modules (or blocks) are depicted in Figure 2.9, with a flowchart that describes the operations performed on data, from its acquisition to the desired regularized segmentation. Decomposing our approach as blocks allows for a generic formulation, with respect to the nature of the problem (reconstruction, denoising, etc.) or the application, and allows methods to be implemented and compared as newer alternatives emerge. Chapter 3 and Chapter 4 introduce the options proposed for those blocks.

Section 3.2 corresponds to the block (i), and presents as a contribution the operators that we gathered and propose to use for constructing maps that indicate the presence and orientation of thin structures of the image. In the block (ii), corresponding to Section 3.3, the challenge is to construct anisotropic neighborhood on the irregular grid of sites, which is the main contribution of our work. Four alternatives for neighborhood construction are proposed in this module, that are all computed from the orientation and saliency maps derived from the previous step. The anisotropic neighborhood constructed allows for finally computing the energy terms of the functional (2.3), functional that is minimized according to the optimization step in block (iii) detailed in Chapter 4.

Chapter 3

Neighborhood Construction

Contents

| | |
|---------------------------------------|-----------|
| 3.1 Isotropic Neighborhood | 20 |
| 3.1.1 Stawiasky's neighborhood | 21 |
| 3.1.2 Disc neighborhood | 21 |
| 3.2 Thin Structures Estimation | 22 |
| 3.2.1 Energy based guidance map | 23 |
| 3.2.2 Tensor Voting | 26 |
| The Tensor Voting framework | 26 |
| Tensor decomposition | 27 |
| Stick kernel | 28 |
| Derivation of other kernels | 30 |
| Voting steps | 31 |
| Feature extraction | 31 |
| 3.2.3 RORPO | 32 |
| The RORPO framework | 32 |
| Dilation | 33 |
| Orientation sampling | 33 |
| Path Opening | 33 |
| The ranking filter | 34 |
| Fine estimation of the orientations | 35 |
| Guidance map construction | 36 |
| 3.3 Anisotropic Neighborhoods | 36 |
| 3.3.1 Shape-based neighborhood | 36 |
| 3.3.2 Dictionary-based neighborhood | 38 |
| 3.3.3 Path-based neighborhood | 39 |
| Target-based neighborhood | 39 |
| Cardinal-based neighborhood | 40 |
| 3.4 Conclusion | 41 |

We have introduced in Chapter 2 the notion of thin structures and motivated the benefice of anisotropic neighborhoods for regularizing those thin structures. In this chapter, we formalize the construction of these neighborhoods, that can be decomposed into the estimation of the presence of thin structures (often performed by a vesselness operator), and the actual computation of the neighbors of each site. In this section, we introduce some of the notations required for doing so.

First, we recall that the neighborhoods are constructed in an irregular lattice of sites, $\mathcal{S} \subset 2^{\mathcal{P}}$, with $\mathcal{P} \subset \mathbb{R}^N$ the set of pixels. In this work the considered sites may either be pixels or superpixels, and we insist on the genericity of our formulation. We denote the cardinal of our sets by $\#$, for instance $\#\mathcal{S}_3$ is the number of sites in our problem.

As introduced previously, it is important to distinguish the *neighborhood*, from what we call *adjacency* and *connectedness* relationship. In this work, the neighborhood refers to the vertices of a graph, that are interconnected with edges. The neighborhood $\mathcal{V} : \mathcal{S} \rightarrow 2^{\mathcal{S}} = \mathbb{V}$ is thus an application that maps the set of sites \mathcal{S} to its powerset without any specific constraint (e.g., bound on spatial distance) at this stage.

Adjacency refers to the existence of a common border between the sites: It is a geometrical observation that is inherent to the topological structure of the sites. Mathematically, adjacency $\mathcal{V} \in \mathbb{V}$ can be defined as follows:

$$\forall s \in \mathcal{S}, \mathcal{V}(s) = \{t \in \mathcal{S} / \exists p \in s, q \in t, \|\overrightarrow{pq}\| = 1\}, \quad (3.1)$$

where $\forall (s, t) \in \mathcal{S}^2, \overrightarrow{st} \in \mathbb{R}^N$ is the vector linking site s and site t . When sites are pixels, this is a trivial definition of a vector between two points of \mathbb{R}^N , and when the sites are superpixels, we compute it as the vector linking the two barycenters of the each superpixels. $\|\cdots\|$ denotes for the norm in \mathbb{R}^N . With this definition, when computing the sites at pixel level, this reduces to the 4-adjacency, since the condition becomes $t \in \mathcal{V}(s) \iff \|\overrightarrow{st}\| = 1$. However, we can modify this constraint in Equation (3.1) for using 8-adjacency or any other model. In practice though, using more elaborated adjacency relationship does not change much the neighborhood since at superpixel level corner contacts are relatively rare, excepted for some algorithms where the topology is constrained.

We note that from the notion of adjacency derives the notion of connected component: A set of elements X is a connected component if any pair of elements $(s, t) \in X^2$ can be connected by pairs of successively adjacent elements, forming a path from s to t . Mathematically, a set $X \subset \mathcal{S}$ is a single connected component if and only if:

$$\forall (s, t) \in X, \exists (p_0, \dots, p_n) \in X^{n+1} \text{ with } n \in \mathbb{N} \text{ such that } \begin{cases} s & \in \mathcal{V}(p_0) \\ p_i & \in \mathcal{V}(p_{i+1}), \forall i \in \llbracket 0, n \rrbracket \\ p_n & \in \mathcal{V}(t) \end{cases} \quad (3.2)$$

Finally *connectedness* in our work deals with sites that are connected in a graph by a structuring function. The latter one may for example be defined for a morphological operator. We denote connectedness $\rightsquigarrow : \mathcal{S} \rightarrow 2^{\mathcal{S}}$. More information will be provided in Section 3.2.3.

The construction of neighborhood and the computation of vesselness operator usually rely on geometrical considerations, like the geometric distance between sites, the angle between vectors, or the projection of a vector on a plane. For this reason, we introduce geometrical notations: The normalized scalar product $\leqslant \cdot, \cdot \geqslant$, that simplifies equations involving angles and projections, defined as follows:

$$\leqslant \mathbf{a}, \mathbf{b} \geqslant = \frac{\langle \mathbf{a}, \mathbf{b} \rangle}{\|\mathbf{a}\| \|\mathbf{b}\|}, \forall \mathbf{a}, \mathbf{b} \in \mathbb{R}^N, \quad (3.3)$$

where $\langle \cdot, \cdot \rangle$ is a scalar product in \mathbb{R}^N , associated with the norm $\|\cdot\|$ such that $\|\mathbf{a}\|^2 = \langle \mathbf{a}, \mathbf{a} \rangle$.

3.1 Isotropic Neighborhood

In this section we introduce the construction of isotropic neighborhoods, with a generic formulation that is valid for both pixels and superpixels.

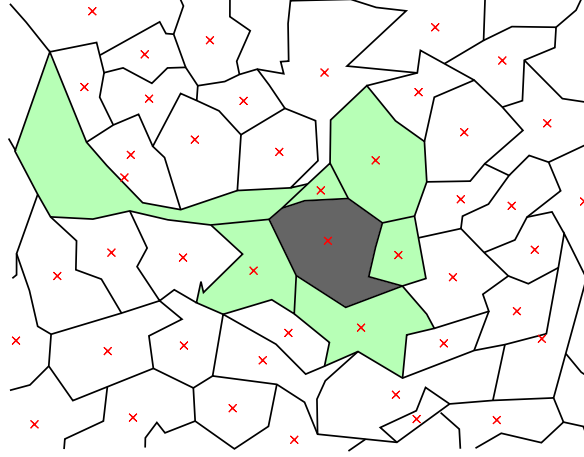


Figure 3.1 – [Stawiaski and Decenci re \[2011\]](#) neighborhood. The neighbors of the reference site (in grey) are pictured in green. We can see that each neighbor is adjacent to the reference one, but some sites may extend spatially far away from their actual neighbor. Therefore, the union of the neighborhood with the site is always a single connected component, but the actual shape of the neighborhood is not controlled. The introduction of constraints dealing with limited sizes and regular shapes of the superpixels may however bring regularity to the shape of the neighborhood.

3.1.1 Stawiasky's neighborhood

[Stawiaski and Decenci re \[2011\]](#) propose a simple way to take advantage of a superpixel segmentation to perform the regularization of an image: The superpixels that share a common border are neighbors, and their interactions are weighted by the length of this common border. As depicted in Figure 3.1, this neighborhood corresponds to the adjacency defined in Section 2.1.2 and Equation (3.1), with a weighting function $W : \mathcal{S}^2 \rightarrow \mathbb{R}$, defined by: $\forall s \in \mathcal{S}, p \in V(s)$:

$$W(s, t) = \sum_{p \in s} \left(\sum_{q \in t} \mathbb{1}_{\{\|\vec{pq}\|=1\}} \right), \quad (3.4)$$

with $\forall s \in \mathcal{S}$:

$$V(s) = \mathcal{V}(s). \quad (3.5)$$

We note that this formulation ensures that the set constituted by a site and its neighborhood is one connected component, but it does not ensure that barycenters of neighboring superpixels are close from each other. Very large sites may therefore be included in the neighborhood of site s while most of the pixels that constitute them are actually far from s site. At pixel level, the formulation of [Stawiaski and Decenci re \[2011\]](#) boils down to an isotropic formulation with the adjacency defined in Equation (3.1).

3.1.2 Disc neighborhood

This formulation, derived from superpatch definition [Giraud et al. \[2017a\]](#), is based on the relative positions of the barycenters of the sites, while disregarding the actual adjacency of sites.

We can mathematically define the neighborhood as: $\forall s \in \mathcal{S}_3$,

$$V(s) = \left\{ t \in \mathcal{S}_3 / \|\vec{st}\| < R \right\}. \quad (3.6)$$

In this neighborhood, there is no constraint of adjacency: A site and its neighbors may not belong to a single connected component. Then, two sites can be neighbors without being adjacent, and vice versa. The fact that the geometric shape of the sites is ignored can sometimes lead to unwanted configurations. However, under some assumptions on the properties of the sites, like some hypotheses on their regularity and compacity for instance, this neighborhood yields consistent results.

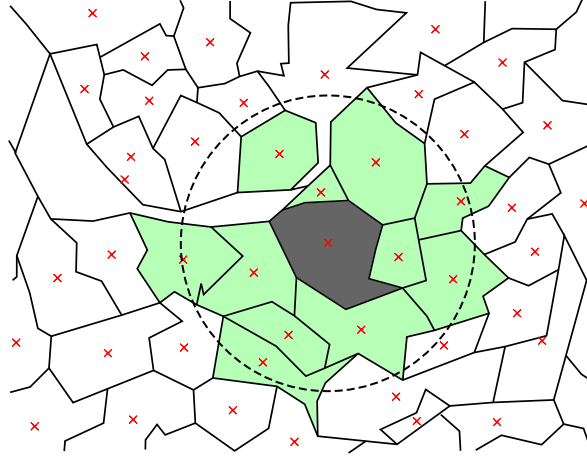


Figure 3.2 – Disc neighborhood. The neighbors of the site s (in grey) are pictured in green. The barycenters are included in a disc centered on the reference site s . The fact that the union of the neighborhood with the site $V(s) \cup \{s\}$ would constitute a single connected component is not ensured, as shown with the green site in the upper left corner, but under some hypothesis of regularity on both convexity and regularity of the sites' area, the likelihood of encountering such configuration is reduced.

3.2 Thin Structures Estimation

When it comes to anisotropic neighborhoods, our formulation will rely on a guidance map that encodes the information of anisotropy and orientation for every site $s \in \mathcal{S}$. Such a map must be sensitive to the structure of the scene to encourage the alignment of neighborhoods with the thin structures of the image. What is expected is that, when a site belongs to a thin structure, its neighbors lie within the structure itself, to ease its good segmentation. The selection of the data to use for computing such a guidance map is not trivial:

- firstly, these data should allow for computing the structures of the image beforehand, and discriminate the thin ones among them,
- secondly, the input data itself may be corrupted, prone to noise, occlusions or for example bad exposure, that can mislead the structure estimation.

One could try to reduce the effects of noise alterations by using regularized data, but obtaining a regularized estimation of the segmentation of the scene yields the question of defining the neighborhood that require a guidance map, leading to a chicken-and-egg situation.

In this situation, we can explore three distinct ways to solve the problem.

- The first one would be to simultaneously estimate the segmentation, the neighborhood and its guidance map. Such an estimation presents the advantage of theoretical optimality, i.e. it should allow for the best segmentation with the best neighborhood according to our model. Unfortunately, the resolution of such a problem is very complex and we will not follow it up in this study.
- The second way is to propose an iterative resolution: the guidance map needs to be initialized at some point, so that the neighborhood can be constructed and a segmentation computed; the latter one is then used for computing an updated guidance map. The advantage of this resolution lies in its simplicity. Its drawback is that it requires that the iterative scheme converges to an optimal segmentation. We present in the following subsection the formulation of a functional with the introduction a single new energy term, that can be minimized iteratively with guaranties of convergence.
- Finally, the third way is an empirical approach of the solution with a single estimation of the guidance map. The methodological and computational advantages of such an approach are

obvious but, to yield some trustworthy guidance map, its estimation should be robust to the input data imperfections that we already mentioned. More specifically, if the guidance map estimation bases on a local estimate of the segmentation, its construction must be robust to noise in the input data, i.e. in the proposed solution to the presence of *False Positive* (FP) and *False Negative* (FN) in the segmentation. For deriving the empirical guidance map, we investigate two options, the *Tensor Voting* (TV) as presented by Medioni et al. [2000] and the *Ranking the Orientation Responses of Path Operators* (RORPO) vesselness operator as introduced by Merveille et al. [2018].

In our works, we implement the guidance map as a vector field $\mathbf{g} \in \mathbb{R}^N$, so that both the direction and the saliency are encoded by a vector, for each site. For computing the values taken by the field, we first propose a formulation based on an energy allowing for an iterative minimization, that we introduce in Section 3.2.1. Then, we present the guidance map construction estimated from TV in Section 3.2.2 and finally we introduce in Section 3.2.3 the use of the RORPO vesselness operator.

3.2.1 Energy based guidance map

The approach of energy based guidance map relies on the computation beforehand of a finite set of configurations of neighbors for each site. The aim is to control the number of available configurations, which is equal to $2^{\#S-1}$ sets of neighbors per site. The computation of the configurations that are selected only takes into account the positions of the sites, not their color, and will be specified in Section 3.3.2. Without additional information, the configurations would be equiprobable. In our case since we aim at designing neighborhoods based on image content, we introduce some hypothesis to preserve thin structures.

The first hypothesis is that if a site belongs to a thin structure, its neighbors should also belong to the same structure. Indeed, since we expect thin structures to be labeled homogeneously, a site should have the same label than its neighbors. This is a strong hypothesis that requires the blind labeling to be correctly estimated. If the labeling is wrong, the neighborhood selection is more likely to fail. For that reason, we introduce a second hypothesis, the smoothness of the neighborhood: Near an anisotropic set of neighbors, we expect to find similar neighborhood configurations. However, the notion of similarity for neighborhood configuration must be discussed as well and could depend of the actual construction of the neighborhood in Section 3.3. Put together, these two hypotheses can be formulated as a problem of regularization of the neighborhood involving two energies to minimize jointly. We illustrate these two hypotheses in Figure 3.3.

In the following of this section we specify these energies.

Our first energy term is derived from our first hypothesis, i.e. the neighbors of a site in a thin structure are expected to lie within the structure itself and the sites in the same structure should share the same label. Therefore, it favors homogeneous labeling between pairs of neighboring sites. One rather intuitive way to formulate this term, in addition to bringing a simple converging minimization scheme, is to consider the regularization term for the labeling in the functional (2.3):

$$E_2(\mathbf{u}, \mathbf{V}) = \sum_{s \in \mathcal{S}} \sum_{t \in V(s)} W(s, t) \mathbb{1}_{\{u_s \neq u_t\}}.$$

With ordered label set, the TVA can be used, and then:

$$E_2(\mathbf{u}, \mathbf{V}) = \sum_{s \in \mathcal{S}} \sum_{t \in V(s)} W(s, t) \|u_s - u_t\|.$$

These terms favor the configurations where the neighbors of a site share the same labeling as this site $s \in \mathcal{S}$. However, in the case of an erroneous label u_s , the orientation of the neighbors would also be erroneous. We have thus also envisaged another form of the energy E_2 that allows for choosing the orientation that maximizes the homogeneity of the neighborhood of s independently of u_s :

$$E(\mathbf{u}, \mathbf{V}) = \sum_{s \in \mathcal{S}} \min_{u \in \mathcal{C}} \sum_{p \in V(s)} W(s, p) \mathbb{1}_{\{u_s \neq u_p\}}.$$

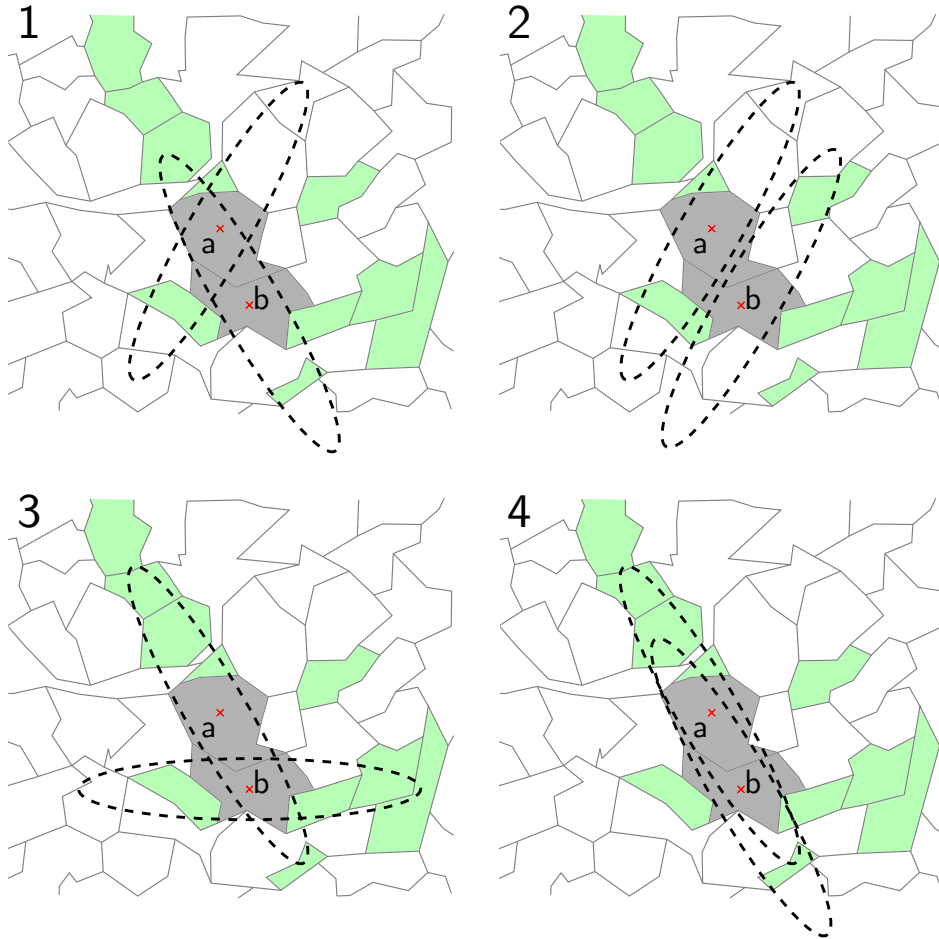


Figure 3.3 – Four examples of neighborhood configurations in which we highlight two sites a and b in grey whereas the sites with label 0 are in white. In the first image, the neighborhoods are oriented randomly, which corresponds to a bad configuration with respect to both hypotheses of labels homogeneity and neighborhood similarity. In second image, the neighborhoods in A and B share the same orientation and are therefore similar, but labels in the neighborhoods are highly heterogeneous and therefore this configuration should be penalized. In third example, both neighborhoods respect the content of the image, but at the expense of an irregularity of the neighborhood field, and finally an acceptable compromise is shown in fourth image in which neighborhoods of A and B are shape-similar and rather label-homogeneous.

Unfortunately, since the sites belonging to a thin structure may be a minority in most of the neighborhoods configurations, this energy rather favors sites belonging to the background. We implemented this energy, and confirmed that, despite the higher sophistication level, such an energy actually reduces the quality of the obtained neighborhood.

These errors in neighborhood estimation motivate us for introducing our second hypothesis about the regularity of the neighborhood shape. Assuming the neighborhood is regular means that spatially “close” sites should share geometrically “similar” neighborhood. For defining similarity, we introduce an energy $E_3(V)$, that encodes the smoothness prior and is therefore a measure of the heterogeneity of the neighborhood.

Since we propose to regularize the neighborhoods themselves, we need to introduce a new functional $F_2 : (\mathcal{C}^S \times \mathbb{V}) \rightarrow \mathbb{R}$, that includes the term E_3 :

$$F_2(\mathbf{u}, V) = E_1(\mathbf{u}) + \alpha E_2(\mathbf{u}, V) + \beta E_3(V). \quad (3.7)$$

This allows for embedding E_3 in a global minimization problem, with $F_2(\mathbf{u}, V) = F(\mathbf{u}, V) + \beta E_3(V)$. The minimization of F_2 has to be performed both with respect to the segmentation \mathbf{u} and to the neighborhood V . Since the only terms depending on \mathbf{u} are in $F(\mathbf{u}, V)$, minimizing F_2 still minimizes F with respect to \mathbf{u} , while also fixing a constraint on the neighborhood V .

Now, we must precise $E_3(V)$ and introduce the measure of dissimilarity between neighborhoods. Since the sites may constitute an irregular lattice, the neighbors of each site may have totally different and unique configuration. For this reason, we introduce the guidance map \mathbf{g} . For any site $s \in \mathcal{S}$, a vector \mathbf{g}_s encodes the orientation and the anisotropy of the set of neighbors $V(s)$, that are respectively represented by the direction and the norm of the vector \mathbf{g}_s . When the neighborhood is totally isotropic, $\mathbf{g}_s = \vec{0}$ and there is no orientation defined. In our approach, V is constructed from \mathbf{g} , as explained in Section 3.3. Then, the fact that V depends on \mathbf{g} by construction allows us to compare the neighborhoods by writing $E_3(V) = E_3(V(\mathbf{g})) = E_3(\mathbf{g}) : \mathbb{R}^N \rightarrow \mathbb{R}$.

We recall that in this section for simplicity, we focus on the case where $N = 2$.

To answer the question of the spatial configuration of the smoothness prior for the neighborhoods, we discretize the guidance map and model it as a MRF. We use the adjacency \mathcal{V} to define the second order cliques of this model. Two neighbor configurations $V(s)$ and $V(t)$ appear in the regularization term only when the corresponding sites s and $t \in \mathcal{S}$ are adjacent.

Being the smoothness term, E_3 is simply written as a sum of elementary pairwise terms $E_3^{st} : \mathbb{R}^N \times \mathbb{R}^N \rightarrow \mathbb{R}$:

$$E_3(\mathbf{g}) = \sum_{s \in \mathcal{S}} \sum_{t \in \mathcal{V}(s)} E_3^{st}(\mathbf{g}_s, \mathbf{g}_t),$$

where, $\forall s, t \in \mathcal{S}$,

$$E_3^{st}(\mathbf{g}_s, \mathbf{g}_t) = \begin{cases} \arccos(|\leq \mathbf{g}_s, \mathbf{g}_t \geq|) & \text{if } \mathbf{g}_s \neq 0 \text{ and } \mathbf{g}_t \neq 0, \\ \arcsin(|\leq \mathbf{g}_s, \vec{st} \geq|) & \text{if } \mathbf{g}_s \neq 0 \text{ and } \mathbf{g}_t = 0, \\ \arcsin(|\leq \vec{st}, \mathbf{g}_t \geq|) & \text{if } \mathbf{g}_s = 0 \text{ and } \mathbf{g}_t \neq 0, \\ 0 & \text{if } \mathbf{g}_s = 0 \text{ and } \mathbf{g}_t = 0. \end{cases} \quad (3.8)$$

These terms represent the angles that we have represented geometrically in Figure 3.4. We recall that $\leq \cdot, \cdot \geq$ stands for the normalized scalar product between two vectors, and thus the term E_3^{st} depends on the angles between \mathbf{g}_s , \mathbf{g}_t , and the orthogonal of \vec{st} . With this definition, E_3^{st} is a pseudo-metric: it is symmetric, respects the triangular inequality and $E_3^{st}(\mathbf{g}_s, \mathbf{g}_s) = 0$. We note that, with this definition, each subproblem associated to the term $E_3(\mathbf{g})$ for α -expansion is submodular, see Appendix B.1.

The only term of the functional F_2 (Equation (3.7)) that depends both of \mathbf{u} and V is $E_2(\mathbf{u}, V)$. Arbitrary fixing \mathbf{u} to an initial value then allows for minimizing the functional with respect to V , and vice versa. When \mathbf{u} is fixed, the minimization of the functional reduces to minimizing $\alpha E_2(\cdot, V) + \beta E_3(V)$. Considering the properties of these two terms with respect to V may allow for minimizing the partial sum of the functional, and finding an intermediate minimizer of the F_2 . We note that the only parameter of this approach is the regularization parameter $\beta \in \mathbb{R}_{\geq 0}$.

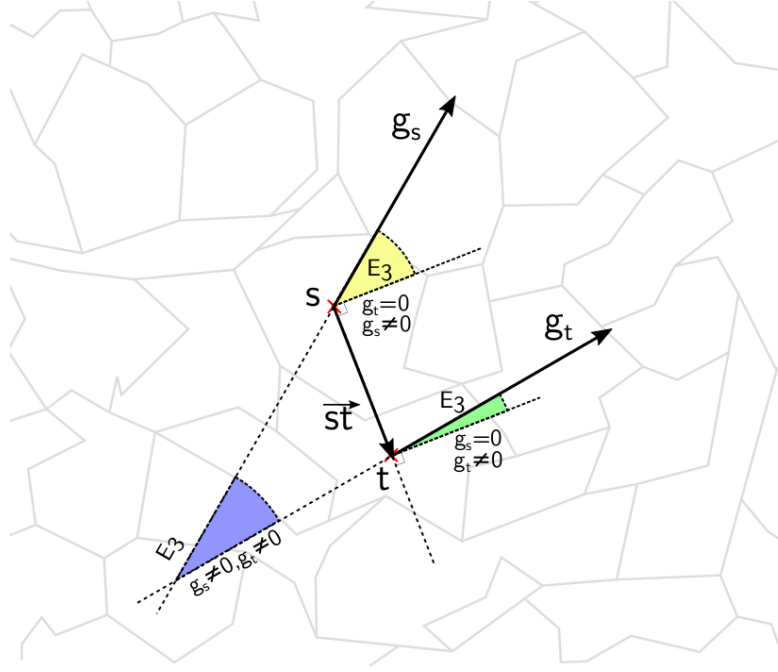


Figure 3.4 – The energy term E_3^{st} (Equation (3.8)) is defined from the angles visualized in color: If one of the guidance map vectors is set to $\vec{0}$ (when the norm of the guidance vector is lower than the anisotropic threshold Γ and thus the neighborhood is considered as isotropic), then the angle considered includes the displacement from s to t . If both the sites are in isotropic configurations, the energy is null, and if both are anisotropic, the energy is symbolized by the blue angle.

The same idea also applies when fixing the value of V . Therefore, the formulation of (3.7) allows for a converging iterative minimization, that we describe later in Section 4.2.

Let us now present the estimation of the guidance map from TV.

3.2.2 Tensor Voting

The Tensor Voting framework

Tensor Voting (TV) has been selected among other solutions for its robustness to noise and efficiency for thin structures like edges [Medioni et al. \[2000\]](#). It has one main scale parameter, $\sigma_T \in \mathbb{R}_{>0}$, that allows for setting the spatial range in which most of the energy of the tensor voting will be distributed. The objective is to refine the estimated orientations of local structures, in order to improve the regularization of thin structures, allowing gap filling without blurring. Actually, tensor voting takes into account the Gestalt principles of perceptual organization (such as proximity, continuity and similarity) for designing the voting operation.

Since TV main interpretation is geometrical, we start introducing 3D TV, and then derive 2D TV from 3D definition which is much simpler than the contrary. Moreover, a presentation in 3D helps to clarify some counter-intuitive conclusions and avoid confusions that may exist in 2D. Therefore, in this section, $N = 3$.

A tensor can be represented by a matrix $\mathbb{T} \in \mathbb{R}^{3 \times 3}$ that has an origin coordinate O in \mathbb{R}^3 and is endowed with a voting function $VF: \mathbb{R}^{3 \times 3} \times \mathbb{R}^3 \mapsto \mathbb{R}^{3 \times 3}$. Casting a vote to other site locations allows the information of each tensor to be propagated in TV. The voting operation VF builds a new tensor \mathbb{T}' to the cast location P and adds it to the tensor at this location, since tensors have good summation properties. The tensor \mathbb{T}' is a combination of rotation and scaling of the source tensor \mathbb{T} , combinations that are all derived from the *stick kernel*. Indeed, tensors can be decomposed in a basis of tensors, in which the stick tensor is the simplest element. The stick kernel refers to the voting operation of this stick tensor.

For presenting the TV algorithm, we therefore start by introducing the tensor decomposition,

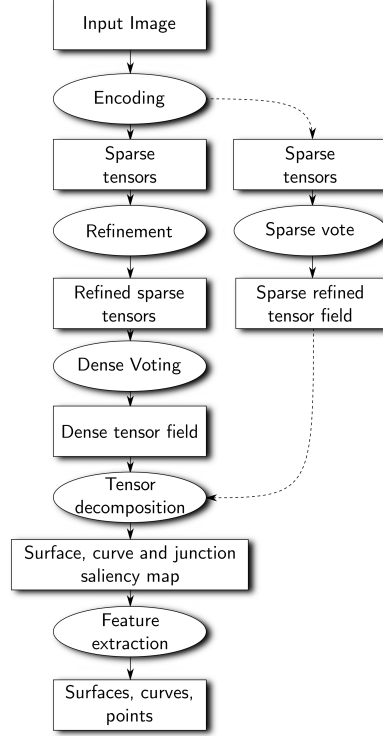


Figure 3.5 – The TV framework in our implementation. Depending on the application, we compute or not a dense tensor map through the dense voting step. In SFF for instance, since the encoding of the initial tensors is sparse in 3D, but dense when projected in 2D, we only do a sparse refinement of the tensors.

then we present the stick kernel, and finally the derivation of the kernels of the other elements of the tensor basis.

The choice of which tensors vote to which location is part of the design of the algorithm. An overview of our implementation, that differs from Medioni et al. [2000] in some aspects, is summarized in Figure 3.5. We discuss these differences in the paragraph about the *voting steps*, and finally we present how tensor map is converted into a guidance map encoding the anisotropy in paragraph *feature extraction*.

Tensor decomposition

Let us introduce some vocabulary relative to the tensors and their representations. In tensor voting, a tensor is a second order symmetric tensor that can be represented by a positive semidefinite diagonalizable matrix $\mathbb{T} \in \mathbb{R}^{3 \times 3}$, whose eigenvectors are orthogonal. We note that its location is important, since the relative position between a tensor and the cast location plays a role in the voting function, and therefore we consider that the three coordinates are a property of each tensor.

In addition to its coordinates, one tensor can be characterized either from six scalar values corresponding to the coefficients of the symmetric matrix or, from three eigenvalues and a rotation. This rotation define the transformation of the orthonormal basis $(\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2)$ to align with $(\hat{\mathbf{e}}_0, \hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2) \in \mathbb{R}^3$, the set of eigenvectors sorted by decreasing eigenvalue. This rotation may either be represented by a unit quaternion (*i.e.* three scalars) or decomposed into three rotations with Euler's angles. Note that both representations are equivalent.

The *decomposition* of the matrix into a set of diagonal matrices is a key point introduced by Medioni et al. [2000]. Let the tensor be a diagonal matrix in the system $(\hat{\mathbf{e}}_0, \hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2)$ (by defini-

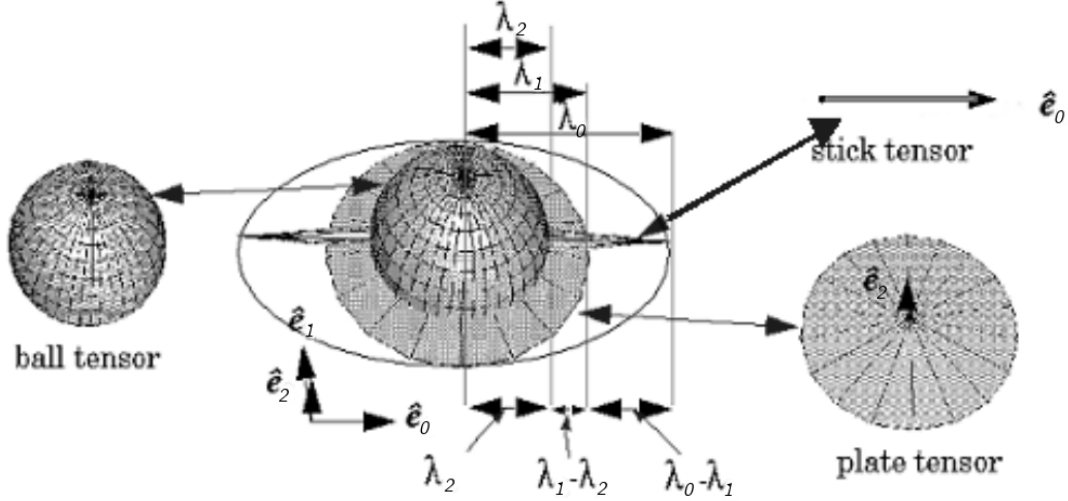


Figure 3.6 – Ellipsoids used for the visual representation of the decomposition of a second-order generic tensor. The stick tensor, with eigenvalue $\hat{\mathbf{e}}_0$, indicates the saliency of surfaces with normal $\hat{\mathbf{e}}_0$. The plate tensor, with a null eigenvalue λ_2 , indicates the saliency of a curve with tangent $\hat{\mathbf{e}}_2$, and the ball component indicates junction saliency. (Image courtesy of Medioni et al. [2005])

tion), and its eigenvalues be denoted $\lambda_0, \lambda_1, \lambda_2$. The decomposition is in the form:

$$\underbrace{\begin{pmatrix} \lambda_0 & 0 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_2 \end{pmatrix}}_{\text{diagonalized Tensor}} = (\lambda_0 - \lambda_1) \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}}_{\text{stick Tensor}} + (\lambda_1 - \lambda_2) \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}}_{\text{plate Tensor}} + \lambda_2 \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\text{ball Tensor}}. \quad (3.9)$$

These elementary tensors are named according to their representations as ellipsoids, as shown in Figure 3.6, and each of them represents a different type of structure as follows. The stick component, written $\hat{\mathbf{e}}_0 \hat{\mathbf{e}}_0^T$, encodes the saliency of surfaces that are normal to $\hat{\mathbf{e}}_0$. The plate component, $\hat{\mathbf{e}}_0 \hat{\mathbf{e}}_0^T + \hat{\mathbf{e}}_1 \hat{\mathbf{e}}_1^T$, is encoding some curves with tangent direction $\hat{\mathbf{e}}_2$. Finally, the ball component is encoding points, e.g. corresponding to thin structure junctions.

Stick kernel

Once the tensors are initialized, voting kernels are designed for allowing TV operation and the spatial propagation of information between tensors. This allows for smoothing the effect of noisy tensors during the voting step and refining their orientations. Since the voting function has an exponential decay with distance from the source tensor, the *voter*, its support is approximated by a finite volume, a sphere, whose radius depends on the scale parameter σ_T . Such a kernel has continuous and smoothly varying orientations of eigenvectors, and smoothly decreasing eigenvalues, excepted at the origin of the kernel. For implementation purpose, the voting kernels are often discretized and stored into a precomputed field of tensors, which evaluates the values of the tensors cast from the voter on each point of a regular lattice, see Section 4.3.3.

Let us now describe the stick kernel as the vote cast by a stick tensor, $\mathbb{T}_{stick} \in \mathbb{R}^{3 \times 3}$.

The stick kernel can be defined by two operations: a multiplication of \mathbb{T}_{stick} by a decay function DF, and a rotation by a vector $\mathbf{\Omega}$. Two geometrical quantities and two constants allow for computing them: the vector \overrightarrow{OP} , between the location of the voter O, and the cast location P, the voter's eigenvector $\hat{\mathbf{e}}_0$, the scale parameter $\sigma_T \in \mathbb{R}_{>0}$ that controls the spatial scale of vote and a constant ν that controls the decay with curvature. Intermediate geometrical quantities that are depicted in Figure 3.7 can be altogether computed from \overrightarrow{OP} and $\hat{\mathbf{e}}_0$.

Among intermediate variables, we find the length of the circle arc $r \in \mathbb{R}_{>0}$ between O and P on the osculating circle joining O and P with normal $\hat{\mathbf{e}}_0$ at point O, and $\phi \in]-\pi, \pi]$ the angle between

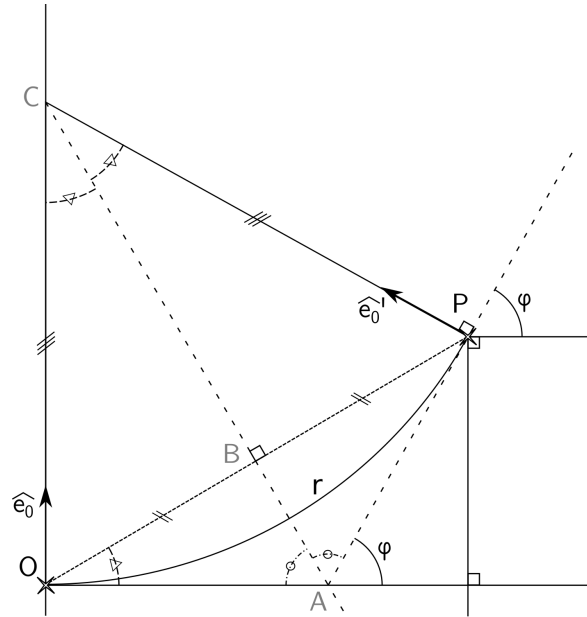


Figure 3.7 – Notations for stick voting kernel. The tensor localized in O with normal $\hat{\mathbf{e}}_0$ casts its vote at point P. The result is a stick tensor which normal is $\hat{\mathbf{e}}'$, with its eigenvalue scaled by the decay function DF. This decay function is computed from the length r of the arc of the osculating circle passing from O to P with normal $\hat{\mathbf{e}}_0$, and from the angle ϕ . All of these geometrical values can be computed from \overrightarrow{OP} and $\hat{\mathbf{e}}_0$.

the tangent to the same osculating circle in O and \overrightarrow{OP} . They can be computed as follows (see Appendix B.2.1):

$$\phi = \pi - 2 \arccos(\leq \hat{\mathbf{e}}_0, \overrightarrow{\mathbf{OP}} \geq),$$

and

$$r = \frac{\pi - 2 \arccos(\leq \hat{\mathbf{e}}_0, \overrightarrow{\mathbf{OP}} \geq)}{4\pi |\leq \hat{\mathbf{e}}_0, \overrightarrow{\mathbf{OP}} \geq|} \|\overrightarrow{\mathbf{OP}}\|$$

Then, the decay function is defined as:

$$\text{DF}(r, \phi, \sigma_{\text{T}}) = \exp\left(-\frac{r^2 + \nu\phi^2}{\sigma_{\text{T}}^2}\right). \quad (3.10)$$

The decay function allows for a smooth voting kernel whose support can be bounded to a sphere of radius $3\sigma_T$. Along with the term $\nu\phi^2$ used for increasing the decay with curvature, Medioni et al. [2000] propose also to restrict vote to the area where $\phi < \frac{\pi}{4}$ and consider that the term $DF(r, \phi, \sigma_T)$ is null otherwise. In two dimensional spaces, this allows for dividing the amount of required calculus by a factor two. In three dimensional spaces, the factor is trickier to calculate but even greater in any case.

The rotation applied to the tensor is such that in the plane containing the cast location P, the location of the tensor O and its eigenvector $\hat{\mathbf{e}}_0$, the new cast tensor's eigenvector $\hat{\mathbf{e}}'_0$ is normal to the osculating circle passing through P and O with normal $\hat{\mathbf{e}}_0$ in O. Geometrically, $\hat{\mathbf{e}}'_0$ and $\hat{\mathbf{e}}_0$ are symmetrical with respect to the mediator of the segment OP:

$$\hat{\mathbf{e}}'_0 = \hat{\mathbf{e}}_0 - 2 \leq \hat{\mathbf{e}}_0, \overrightarrow{\text{OP}} \geq \frac{\overrightarrow{\text{OP}}}{\|\overrightarrow{\text{OP}}\|}.$$

The rotation is defined by the rotation vector $\mathbf{\Omega} \in \mathbb{R}^3$, whose direction is the axis of the rotation and whose norm is the angle of rotation. This rotation transforms the vector $\hat{\mathbf{e}}_0$ into the vector $\hat{\mathbf{e}}'_0$, and is computed as follows:

$$\Omega = \arccos(\langle \hat{\mathbf{e}}_0, \hat{\mathbf{e}}'_0 \rangle) \frac{\hat{\mathbf{e}}_0 \times \hat{\mathbf{e}}'_0}{\|\hat{\mathbf{e}}_0 \times \hat{\mathbf{e}}'_0\|}.$$

This allows for computing the rotation matrix $\mathbf{R}(\boldsymbol{\Omega}) \in \mathbb{R}^{3 \times 3}$ (see Equation (B.4) in appendix). We note that $\mathbf{R}^T(\boldsymbol{\Omega}) = \mathbf{R}^{-1}(\boldsymbol{\Omega}) = \mathbf{R}(-\boldsymbol{\Omega})$, so that the cast tensor $\mathbb{T}'_{stick} \in \mathbb{R}^{3 \times 3}$ can be computed as follows:

$$\mathbb{T}'_{stick} = \text{DF}(r, \phi, \sigma_T) \mathbf{R}(\boldsymbol{\Omega}) \mathbb{T}_{stick} \mathbf{R}^T(\boldsymbol{\Omega}).$$

where \cdot^T is the transposition operation.

With this definition, we obtain the expected smooth stick kernel. This kernel is symmetric with respect to the origin O and invariant by rotation around $\hat{\mathbf{e}}_0$. This kernel is a key parameter for performing the voting steps in TV, since the plate and ball kernels that we introduce in the next paragraph are derived by integration and rotation of the stick kernel.

Derivation of other kernels

For recall, plate tensor can be written $\mathbb{T}_{plate} = \hat{\mathbf{e}}_0 \hat{\mathbf{e}}_0^T + \hat{\mathbf{e}}_1 \hat{\mathbf{e}}_1^T$, while ball tensor is written $\mathbb{T}_{ball} = \hat{\mathbf{e}}_0 \hat{\mathbf{e}}_0^T + \hat{\mathbf{e}}_1 \hat{\mathbf{e}}_1^T + \hat{\mathbf{e}}_2 \hat{\mathbf{e}}_2^T$. The plate and ball kernels derive from the stick kernel by integration of stick tensors. For instance, for plate tensors, the integration of the votes of a stick tensor is done with respect to an angle ρ such that $\mathbb{T}_{stick}(\rho) = \cos(\rho) \hat{\mathbf{e}}_0 \hat{\mathbf{e}}_0^T + \sin(\rho) \hat{\mathbf{e}}_1 \hat{\mathbf{e}}_1^T$.

$$\mathbb{T}'_{plate} = \int_{\rho=0}^{\rho=\Pi} \text{DF}(r, \phi, \sigma_T) \mathbf{R}(\boldsymbol{\Omega}) \mathbb{T}_{stick}(\rho) \mathbf{R}(\boldsymbol{\Omega})^T d\rho.$$

It is important to mention here that the notations have been simplified, but every term under the integral depends on ρ : The decay function DF is computed from the curvilinear abscissa r and ϕ , which in turn are computed from the vector \overrightarrow{OP} and the eigenvector of the stick tensor, $\cos(\rho) \hat{\mathbf{e}}_0 \hat{\mathbf{e}}_0^T + \sin(\rho) \hat{\mathbf{e}}_1 \hat{\mathbf{e}}_1^T$, and the rotation vector $\boldsymbol{\Omega}$, also computed from the same data.

The ball kernel is also defined by integration. The eigenvector of the stick tensor that is integrated is $\hat{\mathbf{e}}_{stick} = \sin(\psi)(\cos(\rho) \hat{\mathbf{e}}_0 + \sin(\rho) \hat{\mathbf{e}}_1) + \cos(\psi) \hat{\mathbf{e}}_2$. The vote of a ball tensor localized in O to the location P is computed as follows:

$$\mathbb{T}'_{ball} = \int_{\rho=0}^{\rho=\Pi} \int_{\psi=-\Pi/2}^{\psi=\Pi/2} \text{DF}(r, \phi, \sigma_T) \mathbf{R}(\boldsymbol{\Omega}) \mathbb{T}_{stick}(\rho, \psi) \mathbf{R}^T(\boldsymbol{\Omega}) \cos(\psi) d\psi d\rho,$$

where r , ϕ and $(\boldsymbol{\Omega})$ all depend both on ψ and ρ .

For numerical computation, we approximate these integrals as sums of tensors as follows:

$$\begin{aligned} \mathbb{T}'_{plate} &\approx \sum_{i=0}^I \text{DF}(r, \phi, \sigma_T) \mathbf{R}(\boldsymbol{\Omega}) \mathbb{T}_{stick}(i\Delta\rho) \mathbf{R}^T(\boldsymbol{\Omega}) \Delta\rho, \\ \mathbb{T}'_{ball} &\approx \sum_{i=0}^I \sum_{j=-J/2}^{J/2} \text{DF}(r, \phi, \sigma_T) \mathbf{R}(\boldsymbol{\Omega}) \mathbb{T}_{stick}(i\Delta\rho, j\Delta\psi) \mathbf{R}^T(\boldsymbol{\Omega}) \sin(j\Delta\psi) \Delta\psi \Delta\rho, \end{aligned}$$

Where $\Delta\rho = \frac{\Pi}{I}$ and $\Delta\psi = \frac{\Pi}{J}$, and $I, J \in \mathbb{N}$ are arbitrary constants. Note that these kernels are often precomputed for computational efficiency.

The analytic expression of the voting function VF for any tensor derives from the notations of the previous paragraphs. Any tensor \mathbb{T}_s at location $s \in \mathbb{R}^3$ can be decomposed from Equation (3.9) in a basis $(\hat{\mathbf{e}}_0, \hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2)$ as $\mathbb{T}(s) = (\lambda_0 - \lambda_1) \hat{\mathbf{e}}_0 \hat{\mathbf{e}}_0^T + (\lambda_1 - \lambda_2) \hat{\mathbf{e}}_1 \hat{\mathbf{e}}_1^T + \lambda_2 \hat{\mathbf{e}}_2 \hat{\mathbf{e}}_2^T$, and the vote cast at location $t \in \mathbb{R}^3$ is written:

$$\begin{aligned} \text{VF}(\mathbb{T}, \vec{st}) &= (\lambda_0 - \lambda_1) \text{VF}(\mathbb{T}_{stick}(t), \vec{st}) + (\lambda_1 - \lambda_2) \text{VF}(\mathbb{T}_{plate}(t), \vec{st}) + \lambda_2 \text{VF}(\mathbb{T}_{ball}(t), \vec{st}) \\ &= (\lambda_0 - \lambda_1) \text{DF}(r, \phi, \sigma_T) \mathbf{R}(\boldsymbol{\Omega}) \mathbb{T}_{stick} \mathbf{R}^T(\boldsymbol{\Omega}) \\ &\quad + (\lambda_1 - \lambda_2) \int_{\rho=0}^{\rho=\Pi} \text{DF}(r, \phi, \sigma_T) \mathbf{R}(\boldsymbol{\Omega}) \mathbb{T}_{stick}(\rho) \mathbf{R}(\boldsymbol{\Omega})^T d\rho \\ &\quad + \lambda_2 \int_{\rho=0}^{\rho=\Pi} \int_{\psi=-\Pi/2}^{\psi=\Pi/2} \text{DF}(r, \phi, \sigma_T) \mathbf{R}(\boldsymbol{\Omega}) \mathbb{T}_{stick}(\rho, \psi) \mathbf{R}^T(\boldsymbol{\Omega}) \cos(\psi) d\psi d\rho. \end{aligned} \tag{3.11}$$

Voting steps

Now that we have introduced the voting operation for one tensor, we detail how these voting operations are used for detecting thin structures.

Two sets $\mathcal{S}_0, \mathcal{S}_1 \subset \mathcal{S}$ are defined respectively to be the set of voters and the set of cast locations (for vote). For instance, if the vote is dense then $\mathcal{S}_1 = \mathcal{S}$. $\forall s \in \mathcal{S}$, let $\mathbb{T}(s)$ be the tensor at location s before the vote, $\mathbb{T}'(s)$ the tensor at location s after vote, and $\text{VF} : (\mathbb{R}^{3 \times 3} \times \mathbb{R}^3) \mapsto \mathbb{R}^{3 \times 3}$ the voting function that returns the tensor cast at location. The voting operation is computed as follows:

$$\begin{cases} \forall p \notin \mathcal{S}_1, & \mathbb{T}'(p) = \mathbb{T}(p), \\ \forall p \in \mathcal{S}_1, & \mathbb{T}'(p) = \mathbb{T}(p) + \sum_{s \in \mathcal{S}_0} \text{VF}(\mathbb{T}(s), \vec{s}p), \end{cases}$$

[Medioni et al. \[2000\]](#) depicts the tensor voting as a five step process (see Figure 3.5). First step is input encoding. In this step, the idea is to select a set of sites of interest, that initializes the set of tensors that will vote first. In literature, the elements of this set are often called *tokens*. Because the objective is to propagate information from each token in order to infer the existence of structures that may connect them, the set of tokens is converted into a sparse set of ball tensors.

Second step is the refinement step, thanks to a sparse vote that allows for refining the initial set of ball tensors into a set of stick tensors. The refinement operation is a projection of each tensor on the stick tensor axis in the basis used for tensor decomposition. It corresponds to removing the plate tensor and the ball tensor component, and keeping for each tensor the stick component oriented in the direction of $\hat{\mathbf{e}}_0$.

Third step is a dense voting in \mathbb{R}^N in order to propagate the stick information in every point: every computed tensor votes on any available site in \mathcal{S} .

We have found in practice that our implementation had to differ from [Medioni et al. \[2000\]](#) in some of these first three steps, to allow us for computing our algorithm in a reasonable time. The non dense voting step of refinement of the tensors for example may be skipped for specific application-related reasons, or handled instead by a specific initialization. Depending on the application, we can initialize stick, plate or ball tensors, such that the initialization tensors can encode multiple information: respectively the presence of a plane, a curve or a point in input data at the tensor's location. Similarly to [Hariharan and Herfet \[2018\]](#) that remove the refinement step and use plate tensors along the optical axis \mathbf{e}_2 , we instead remove the dense voting step and adapt our feature extraction to match with our application. We depict this difference later in application related chapters, Section 6.2.3.

Feature extraction

Once the tensor map is computed, the final steps aim at extracting information so that we can compute the guidance map $\mathbf{g} \in \mathbb{R}^{3^{\mathcal{S}}}$.

Originally, [Medioni et al. \[2000\]](#) project tensors on the three axes of the decomposition basis so that three saliency maps can be derived, encoding for surface, curve and junction saliency. From these maps, the final step of the algorithm aims at deriving the probabilities of presence of surfaces, curves and points. For instance, for plane detection, the analysis focuses on the stick component.

In our implementation, we also compute a tensor decomposition as introduced in Equation (3.9). The saliency and orientation of the guidance map are computed respectively from the stick saliency and the eigenvalues orientation. Conversely to [Medioni et al. \[2000\]](#), we are not interested at this stage by localizing precisely our structures, since we will detect them later from the regularized segmentation. The structure orientation is defined as follows, $\forall s \in \mathcal{S}$,

$$\mathbf{g}_s = (\lambda_{0s} - \lambda_{1s})\hat{\mathbf{e}}_{0s}. \quad (3.12)$$

When working in a 3D application, such as SFF, we need to adapt this definition so that the guidance map is usable for 2D neighborhoods. The question of computing the set of 3D tensors for obtaining a 2D guidance map will be addressed in Section 6.2.3.

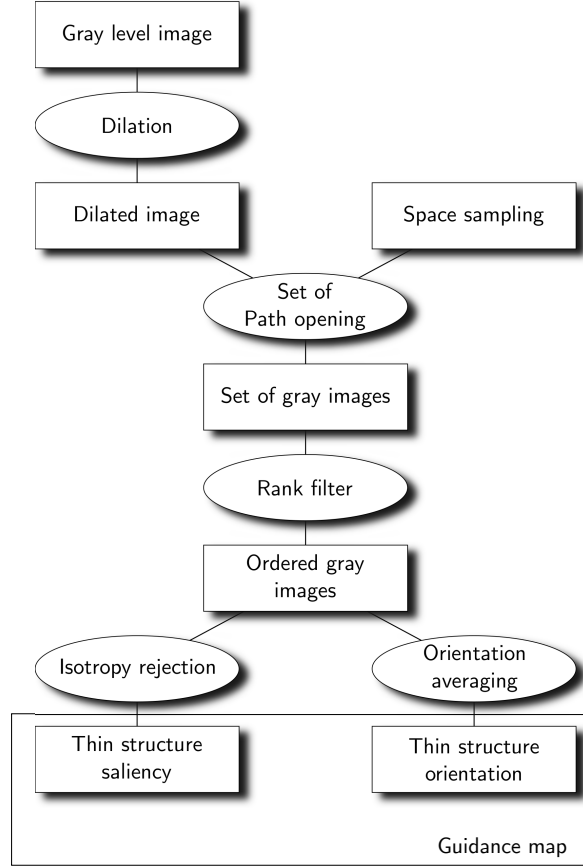


Figure 3.8 – The implementation of the RORPO framework.

3.2.3 RORPO

The RORPO framework

As an alternative to TV, we consider RORPO proposed by Merveille et al. [2018], which is a non linear operator based on mathematical morphology and used for thin structure detection. For our implementation, we compute the RORPO in 2D because we only have 2D neighborhoods. However, since with SFF, the dataset is a stack of 2D images, we compute RORPO on each 2D image separately and combine the results afterwards in a specific operation described in Section 6.2.3. The idea of RORPO is to use a set of oriented filters to perform multiple morphological operations on the same image, but with different orientations. Then, the number of “high” responses in this set of filters is counted to distinguish the thin structures from the isotropic areas. By definition, at least one dimension of a thin structure is substantially smaller than the other ones. Thus, finding the sites where only a small number of high responses are measured among the oriented filters discriminates the thin structures.

These oriented filters are morphological operations called path openings. A path opening is based on the notion of adjacency of the sites (\mathcal{V}), for an anisotropic structuring function. This structuring function defines a set of non-symmetric connection relationships \mathcal{R} with a preferred orientation, and the set of considered structuring functions achieves a sampling of the possible orientations of the image plane. Compared to TV, RORPO allows for a faster computation of the guidance map and is consistent with the notion of path-based neighborhood introduced in Section 3.3.3. Since in our applications we only consider 2D neighborhoods, let us focus on 2D adjacency \mathcal{V} between elements (i.e, sites).

Figure 3.8 describes the RORPO implementation: The input is a grayscale 2D map that is first dilated with respect to its spatial adjacency (cf. just below). We sample the orientations of the 2D space so that only a finite set of structuring functions are used for performing path openings.

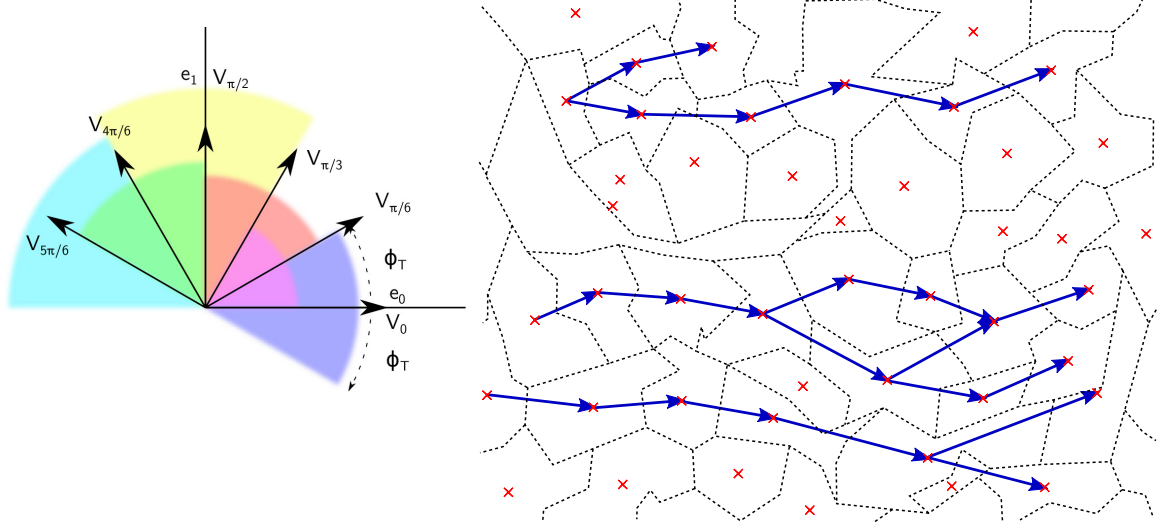


Figure 3.9 – Illustration of the 6 directions of $\mathcal{S}_{\rightsquigarrow}$ (left) and an example of path obtained with one structuring function \rightsquigarrow_θ (right). The connectedness \rightsquigarrow_θ is characterized by the vector \mathbf{v}_θ and the angular width ϕ_T . For this illustration, we have represented directed edges for positive displacements, but the paths are computed in both directions.

Then, the responses of the path operators are sorted, and compared, to preserve only structures that have a small number of high responses in the set of path openings. As a final step, a site-wise minimum is computed between the initial grayscale image and the intermediate RORPO filter's answer in order to preserve the anti-extensivity of the path opening.

We detail these steps in the following of this section.

Dilation

In RORPO, the site-wise dilation of the grey level input image $X' \in \mathbb{R}^N$ is defined as follows: $\forall s \in \mathcal{S}$,

$$X(s) = \max_{p \in \mathcal{V}(s) \cup \{s\}} (X'(p)), \quad (3.13)$$

where $\mathcal{V}(s) \subset \mathcal{S}$ is the set of sites adjacent to the site $s \in \mathcal{S}$, which means that they share a common border (see definition Equation (3.1)).

Orientation sampling

Next step deals with direction sampling. For our application we consider 6 directions \mathbf{v}_θ in the image plane, characterized by their positive angle θ with the \mathbf{e}_0 axis: $\mathbf{v}_\theta = \cos(\theta)\mathbf{e}_0 + \sin(\theta)\mathbf{e}_1$ as shown on Figure 3.9. Next step is path computation that is based on a oriented (irreflexive and non-symmetric) structuring function. In our case, not specifying the nature (pixels versus super-pixels) of the sites of \mathcal{S} , we base the anisotropic connections on the relative positions of the sites barycenters. The connection relationship between sites s and t , denoted $s \rightsquigarrow_\theta t$, is defined as follows:

$$\forall (s, t) \in \mathcal{S}, \left\{ s \rightsquigarrow_\theta t \iff t \in \mathcal{V}(s) \text{ and } \left| \langle \vec{st}, \mathbf{v}_\theta \rangle \right| > \cos(\phi_T) \|\vec{st}\| \right\},$$

where ϕ_T represents the angle threshold of the connection relationship, as illustrated in Figure 3.9. In the following, \mathcal{R} denotes the set of connection relationships.

Path Opening

For each connection relationship $\rightsquigarrow_\theta \in \mathcal{R}$, a site-wise gray-level path opening is an extension of the binary path opening defined in Merveille et al. [2018].

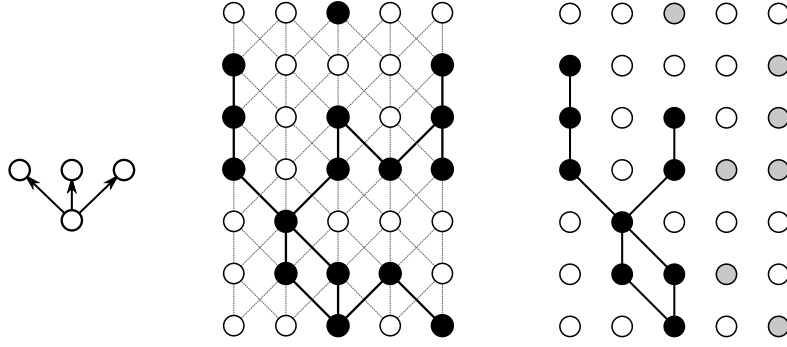


Figure 3.10 – Example of path opening of length 4 with the connection relationship \rightsquigarrow_θ ($\theta = \frac{\pi}{2}$) depicted on the left hand side. The middle image is a the binary input image where the \rightsquigarrow_θ links between active sites are shown in bold. On the right hand side we have performed the path opening, and consequently all the paths of length below 4 are removed.

Let us first consider the case of a binary image X and let $\tilde{X} \subset \mathcal{S}$ be the set of *true* (or 1-valued) pixels in X . \tilde{X} is called the support of X . Given the connection relationship \rightsquigarrow_θ and a length $L \in \mathbb{R}_{>0}$, the path opening $\mathbb{O}_{\rightsquigarrow_\theta}(L, \tilde{X})$ is the union of all paths connected by \rightsquigarrow_θ of length L in \tilde{X} . This operator preserves each point of \tilde{X} belonging to at least one path of $\mathbb{O}_{\rightsquigarrow_\theta}(L, \tilde{X})$, and removes the others, as shown in Figure 3.10.

$$\mathbb{O}_{\rightsquigarrow_\theta}(L, \tilde{X}) = \left\{ \begin{array}{l} s \in \tilde{X} \subset \mathcal{S}, \text{ such that:} \\ \exists (p_0, \dots, p_k) \in \tilde{X}^{k+1}, \text{ with } k \in \mathbb{N} \text{ and :} \\ \forall i \in \llbracket 0, k \rrbracket, p_i \in \mathcal{V}(p_{i+1}), \\ \sum_{0 \leq i < k} (\|p_i p_{i+1}\|) \geq L, \text{ and} \\ p_0 = s. \end{array} \right\}$$

Note that, although usually the length L is a positive integer expressed in pixel unit, extending the case of pixel lattice to superpixel is not trivial. We instead consider that the length of the path is a real $L \in \mathbb{R}_{>0}$, computed as the sum of the distances between the sites' barycenters in the path.

The case of a gray-level image Y is handled through level sets: We denote by $X_{\geq \tau}$ the binary image having *true* values in sites with gray level greater than τ in Y . Let $\tilde{X}_{\geq \tau}$ be the support of $X_{\geq \tau}$, also called level set of value τ . Given the connection relationship $\rightsquigarrow_\theta \in \mathcal{R}$ and a length $L \in \mathbb{R}_{>0}$, the gray-level opening of Y is defined as:

$$\mathbb{G}_{Y, \rightsquigarrow_\theta, L}(s) = \max\{\tau \in \mathbb{R}_{>0} | s \in \mathbb{O}_{\rightsquigarrow_\theta}(L, \tilde{X}_{\geq \tau})\}.$$

Since we consider $\#\mathcal{R}$ connection relationships, each one leading to a path opening result, considering the set of connection relationship, we get a total of $\#\mathcal{R}$ gray-level path openings.

The ranking filter

Each of these path openings filters out the structures that are not aligned with a specific orientation. By definition, the curvilinear structures can only be preserved by a limited number of filters since they are thin structures. This means that at least one oriented filter will delete the structure. Conversely, isotropic structures will have homogeneous answer to the set of path openings.

Therefore, we use a non linear filtering technique named rank filtering that is as follows: For each site $s \in \mathcal{S}$, the responses to the $\#\mathcal{R}$ path openings are ranked according to decreasing order of magnitude. Then, we denote RF_1 the maximum value, $\text{RF}_{\lfloor \#\mathcal{R}/2 \rfloor}$ the median value and $\text{RF}_{\#\mathcal{R}}$ the minimum (last in the ordering) value, with $\lfloor \cdot \rfloor$ denoting the floor function. This ranking of the orientation responses of the path openings gave its name to the algorithm RORPO.

Let $i_{\mathcal{R}} \in \llbracket 1, \#\mathcal{R} - 1 \rrbracket$ denote a threshold value set to the maximum number of high responses expected for a thin structure in our set of path openings. It depends on the orientation sampling and is empirically set to discriminate the isotropic structures from the thin structures.

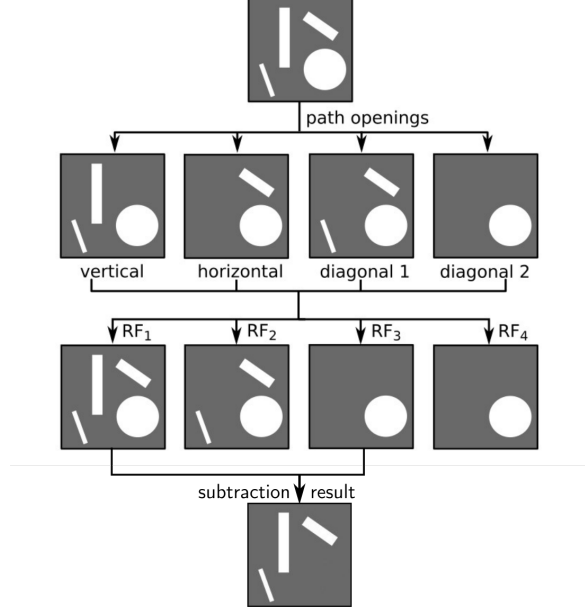


Figure 3.11 – Illustration of the behavior of the ranking filter on the path openings, and the effect of the subtraction method used for only preserving thin structures. We can see that the circle, preserved by all the openings, is removed in the final outcome (courtesy of Merveille et al. [2018]).

For each site, the derived $i_{\mathcal{R}}^{th}$ value is then compared to the maximal one. Now, since a dilation is performed on input image Y , while RORPO operator is expected to be anti-extensive, Merveille et al. [2018] apply an infimum operator to the final RORPO result:

$$\text{RORPO}_Y(s) = \min \left(\text{RF}_1 [\rightsquigarrow_{\theta} \in \mathcal{R}, \mathbb{G}_{Y, \rightsquigarrow_{\theta}, L}(s)] - \text{RF}_{i_{\mathcal{R}}} [\rightsquigarrow_{\theta} \in \mathcal{R}, \mathbb{G}_{Y, \rightsquigarrow_{\theta}, L}(s)], Y(s) \right)$$

Note that if the thin structures of interest present low values with respect to the background, we invert the order of the values before applying RORPO.

This formulation yields higher responses for thin structures that yield a small number of high responses in path openings. Therefore, the value returned by the RORPO allows us to discriminate the saliency of thin structures. We illustrate this in Figure 3.11, where we can see that only the thin structures preserved by less than two path openings are actually present in the final outcome of RORPO.

Let us now present the estimation of orientations used to derive the guidance map \mathbf{g} .

Fine estimation of the orientations

The RORPO operator allows for deriving, for each site, an orientation. For each site $s \in \mathcal{S}$, Merveille et al. [2018] extract this information by averaging the orientations of the $i_{\mathcal{R}} - 1$ largest responses given by the structuring functions of the path openings $\mathbb{G}_{Y, \rightsquigarrow_{\theta}, L}(s)$. We proceed to this averaging as well but in order to ensure appropriate averaging, we must apply a transformation in 2D so that orthogonal vectors cancel themselves and vectors of opposite directions would not.

Because our search for thin structures is limited to 2D space, the set of directions in \mathcal{R} is encoded by a single scalar θ , that is the angle between the direction of the vector \mathbf{v}_{θ} and the reference \mathbf{e}_0 . If we project the problem in the complex plane, or in polar coordinates, the transformation we use is equivalent to doubling the argument value, i.e. the polar coordinate, of the vectors before averaging them, and dividing the argument of the averaged result by two.

Mathematically, with complex notations, we can write our orientation averaging as follows:

$$\mathbf{v}_{\text{RORPO}}(s) = \Re(s) \left(\epsilon_R + \sum_{\rightsquigarrow_{\theta} \in \mathcal{R}'(s)} \mathbb{G}_{Y, \rightsquigarrow_{\theta}, L}(s) \exp(2i\theta) \right)^{1/2},$$

where $\aleph(s) \in \mathbb{R}_{>0}$ is a coefficient that normalizes the vector $\mathbf{v}_{\text{RORPO}}(s)$. The term $\mathcal{R}'(s) \subset \mathcal{R}$ is the set of orientations of the $(i_{\mathcal{R}} - 1)$ first answers in the rank filter at site s . For recall, the vectors \mathbf{v}_θ in complex notations are written:

$$\mathbf{v}_\theta = \exp(i\theta). \quad (3.14)$$

With this construction we obtain a field of vectors that has the same dimensions as the grayscale input data X' , for instance a 2D image in crack detection or a 3D volume with SFF.

Guidance map construction

With 2D data, the guidance map is computed from the RORPO's output with

$$\mathbf{g}_s = \text{RORPO}(s) \mathbf{v}_{\text{RORPO}}(s). \quad (3.15)$$

Note that the application-dependent issue of dealing with 3D data for 2D neighborhoods requiring a 2D guidance map is addressed in Section 6.2.3. Now, we detail more how the actual neighborhoods are built from the guidance map that we have finally constructed.

3.3 Anisotropic Neighborhoods

This section depicts our contribution concerning the construction of anisotropic neighborhoods. The fact that our problem may deal with sites that do not form a regular lattice and the constraint to deal with thin structures motivates our new anisotropic neighborhood formulation. In the following subsections, we aim at defining the neighborhood field $V : \mathcal{S} \rightarrow 2^{\mathcal{S}} = \mathbb{V}$, possibly anisotropic and non stationary. For anisotropic cases, the guidance map $\mathbf{g} \in \mathbb{R}^{2^{\mathcal{S}}}$ contains the data about the orientation and saliency of the structures in the scene, and is therefore used for constructing the neighborhood. Note that, for any site $s \in \mathcal{S}$, when the norm of the guidance vector is below a given threshold $\|\mathbf{g}_s\| < \Gamma$, we consider isotropic neighborhood.

While our definitions could be used for 3D neighborhoods, we note that our application involve 2D neighborhoods in the plane $(\mathbf{e}_0, \mathbf{e}_1)$. Thus, the neighbors of a site are only computed among the sites at a constant depth (in SFF application), i.e. in the same plane. For notation simplicity, we therefore introduce neighborhoods in 2D context.

3.3.1 Shape-based neighborhood

Shape-based neighborhood are computed from the coordinates of the sites' centroids and are independent of their boundaries. The notion of "shape" derives from the representation of the smallest subspace of \mathbb{R}^2 containing all the centroids of the sites that are neighbor of a site whose centroid is taken as the origin. We illustrate this idea with the neighbors of a site for an elliptic shape in Figure 3.12. We note that with this neighborhood, the radiometry of the sites (represented in gray level in the figure at pixel level) does not impact in any way the neighborhood construction.

Let $\mathcal{S}_s \subset \mathbb{R}^2$ be a shape defining the neighborhood at s , $\forall s \in \mathcal{S}$. In the general case, the shape-based neighborhood of each site is defined as follows:

$$V(s) = \left\{ t \in \mathcal{S} / \vec{st} \in \mathcal{S}_s \right\}.$$

In previous equation, since \mathcal{S}_s is a subset of \mathbb{R}^2 , we say that a vector belongs to \mathcal{S}_s as soon as its coordinates do.

We note that the properties of the shapes $(\mathcal{S}_s)_{s \in \mathcal{S}}$ impact the properties of the neighborhood: for the shape of the set of neighbors to be regular, we expect \mathcal{S}_s to be rather compact and convex.

We considered in our case parametric ellipses, that boil down to disks in isotropic situations. For any $s \in \mathcal{S}$, let $\epsilon \in [0, 1[$ be the ellipse eccentricity, and let $A_V \in \mathbb{R}$ be the area of the shape, the relation $\vec{st} \in \mathcal{S}_s$ becomes:

$$(1 - \epsilon^2) \leq \vec{st}, \mathbf{g}_s \geq^2 + \left\| \vec{st} - \frac{\vec{st}, \mathbf{g}_s}{\|\mathbf{g}_s\|} \mathbf{g}_s \right\|^2 < \sqrt{1 - \epsilon^2} \frac{A_V}{\pi}, \quad (3.16)$$

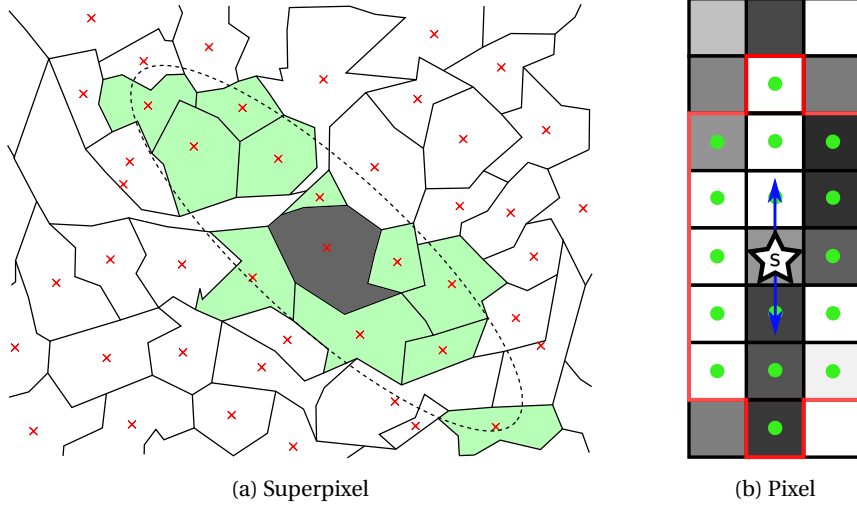


Figure 3.12 – Example of shape-based neighborhood: At superpixel level (a), the shape is represented by the dotted line centered on the grey site. The neighbors of this site are the ones whose centroid (red cross) belong to that shape, and are colored in green. At pixel level (b) the shape is represented by a red continuous line centered on the site s represented by a star. The neighbors of s are all the sites within the shape, indicated by a green dot. We note that the radiometry of the sites (represented in grey level) does not impact in any way the neighborhood construction.

where \mathbf{g}_s is given by the guidance map, aligned with the major semi axis of the ellipses, \vec{st} is the vector joining s and t , and $\leq \cdot, \cdot \geq$ denotes the normalized scalar product. When $\|\mathbf{g}_s\| = 0$, this expression is not numerically stable. However, a null guidance vector corresponds to a null saliency at this location. To detect the cases where isotropic shape (and thus neighborhood) is best suited, we set an isotropic threshold $\Gamma \in \mathbb{R}_{>0}$ on below which the isotropic neighborhood is chosen. The relation $\vec{st} \in \mathcal{S}_s$ then is equivalent to:

$$\|\vec{st}\|^2 \leq \frac{\Delta_V}{\pi},$$

since the elliptic shape becomes a disc. When $\mathbf{g}_s < \Gamma$, the shape-based neighborhood boils down to defining the neighborhood of a site as the isotropic superpatch of Giraud et al. [2017a].

We note that the ellipse eccentricity ϵ may either be deduced as a function of $\|\mathbf{g}_s\|$ or fixed as a constant parameter. On the one hand, having a variable eccentricity value allows for a better flexibility of the neighborhood, since it will depend on the saliency of the site. On the other hand, the high sensitivity of $\|\mathbf{g}_s\|$ to both the image topology and the parameters and model used for estimating it (in our case energy-based, TV or RORPO-based models) brings a lot of issues and complexity when it comes to evaluating separately the benefits of each guidance map model and anisotropic neighborhood. Therefore, we ended up considering ϵ as a constant parameter of the model and decided to leave the problem of a finer dynamic estimation of its value open for future works.

Geometrically, the shape-based neighborhood only depends on the relative positions of the sites (their barycenters), and on the guidance map. When it comes to superpixels, such neighborhood has the same limitations as its isotropic equivalent, the disc neighborhood (Section 3.1.2): Since they are computed from barycenters positions, they do not enforce adjacency of the neighbors. In particular, when the superpixels are highly irregular, concave, or with low compactness, they very likely produce neighborhoods with disconnected components and that may be far from the original site.

However, with compact, regular and convex superpixels, this neighborhood provides an efficient and intuitive method for building anisotropic neighborhoods.

Since the number of sites is finite, the number of configurations that this neighborhood can produce with an arbitrary guidance map is finite. However, if the computation of these configurations is required beforehand, this number can be highly dissuasive with large shapes, since

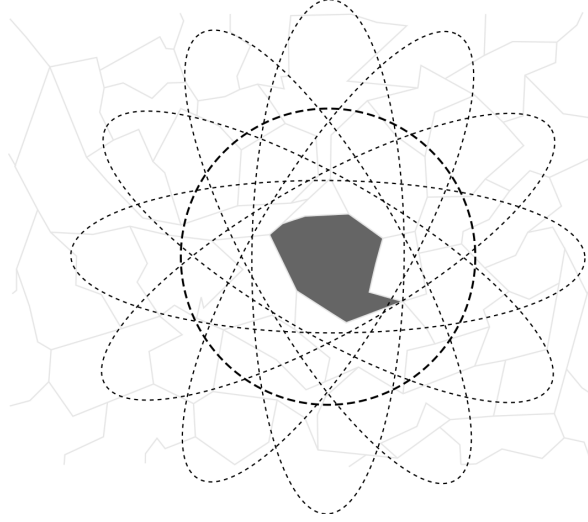


Figure 3.13 – The dictionary neighborhood defines a set of available configurations beforehand. In the case of a shape neighborhood, it corresponds to a set of shapes among which the optimization algorithm has to select a best candidate. Here, the six ellipses aim at capturing most anisotropic thin structures while a disc is used for isotropic regularization of homogeneous regions.

that in the general case the values that can take \mathbf{g}_s are not discretized. Therefore, we introduce the dictionary-based neighborhood, that allows for the practical computation of the energy-based guidance map presented in Section 3.2.1.

3.3.2 Dictionary-based neighborhood

When the values of the guidance map are discrete and finite, the neighborhood is called dictionary-based. This name comes from the fact that the set of shapes that derives from the finite set of values of the guidance map forms a dictionary of shapes. There is therefore a limited and constant number of configurations available, that can be precomputed. This is particularly important for energy-based guidance map described on Section 3.2.1, that requires such precomputed set of neighbors in order to compute the lowest energy configuration possible for the whole neighborhood. The bigger the dictionary, the more fitted the neighborhood, but such a flexibility is obtained at the trade of an increased computational complexity.

The dictionary-based neighborhood that we implemented is very similar to the shape-based neighborhood. The dictionary denoted by \mathcal{D} is composed of ι shapes. In our experiments, $\iota = 7$: Six ellipses account for six possible orientations of the guidance map when the neighborhood is anisotropic ($\mathbf{g}_s > \Gamma$) plus the isotropic disc otherwise. We illustrate this dictionary in Figure 3.13, while an example of associated neighborhood is shown in Figure 3.12.

As in Section 3.3.1, for a fixed shape, the barycenter positions are the only information that determine whether a site has another site as a neighbor or not:

$$V(s) = \left\{ t \in \mathcal{S} / \overrightarrow{st} \in \mathcal{S}_s \in \mathcal{D} \right\}.$$

With a discrete and finite set of configurations per site, we can index the configurations by an integer and therefore define a field of label in the energy formulation. Practically this is the retained implementation for our energy minimization algorithm.

We note that even if we have only implemented the shape-based version of the dictionary-based neighborhood, nothing prevents us from also discretizing the guidance map for path-based neighborhoods that we introduce in the next section, thus implementing an energy-based guidance map suitable for dictionary of path-based neighborhood.

Both for enforcing the set of neighbors (plus the source site) to form a single connected component and for allowing the neighborhood to follow thin structures in the image, we introduced path-based neighborhoods.

3.3.3 Path-based neighborhood

Path-based neighborhoods are designed to fit into thin structures of the image, possibly one site width, especially when working with superpixels.

We use the adjacency graph to build paths that define a set of neighbors. The selection of the optimal path is done by introducing some constraints. In our case, as we want to ensure that the neighbors of a site define a single connected component with the site, we build the neighborhood from the adjacency \mathcal{V} . As a reminder, two sites are adjacent when they share a common border. The notion of border derives from 4-adjacency at pixel level, thus the adjacency is a symmetric relationship ($s \in \mathcal{V}(t) \iff t \in \mathcal{V}(s), \forall s, t \in \mathcal{V}$).

A path of length $n \in \mathbb{N}$ is an ordered list of sites (s_0, \dots, s_n) such that $\forall i \in \llbracket 0, n \rrbracket, s_{i+1} \in \mathcal{V}(s_i)$. We denote the set of paths joining two sites $s, t \in \mathcal{S}$ by $\Pi_\delta(s, t) \subset 2^{\mathcal{S}}$. This set gathers the paths of any length joining the two sites, without loops, and implicitly involves the notion of adjacency \mathcal{V} , since consecutive sites in a path are adjacent. The set of paths of length $K \in \mathbb{N}$ is denoted $\Pi_{\delta K}(s, t) \subset \mathcal{S}^K$, such that:

$$\Pi_{\delta K}(s, t) = \left\{ (s_0, \dots, s_K) \in \mathcal{S}^K / \begin{array}{l} \forall k \in \llbracket 0, K \rrbracket, s_{k+1} \in \mathcal{V}(s_k), s_0 = s, \text{ and } s_K = t \\ \forall j, k \in \llbracket 0, K \rrbracket, s_j \neq s_k \end{array} \right\}.$$

By extension, the set of paths of all lengths can be defined as:

$$\Pi_\delta(s, t) = \cup_{K \in \mathbb{N}} \Pi_{\delta K}(s, t).$$

The path-based neighborhood is computed from the guidance map $\mathbf{g} \in \mathbb{R}^{2\mathcal{S}}$ which defines an orientation in each site. When the norm of the guidance map in a given site s is below a fixed threshold ($\|\mathbf{g}_s\| < \Gamma$), the neighborhood in s is $V(s) = \mathcal{V}(s)$, an isotropic neighborhood that ensures adjacency. Otherwise, the set of neighbors is given by the union of the elements of two paths that expand from the site s to the two opposite directions corresponding to the orientation of \mathbf{g}_s .

A major difference with the shape-based neighborhoods presented in Section 3.3.1, is that the selection of the optimal paths does not depend only on the sites' barycenter positions, but also take into account the site adjacency and, as we will see further, the input data such as radiometry. We present now two options for constructing the path-based neighborhoods.

Target-based neighborhood

Target-based neighborhood is derived from paths that join pairs of distant sites $t_0^*, t_1^* \in \mathcal{S}$ (named "target") for each source site $s \in \mathcal{S}$. The adjacency along these paths is ensured by construction. We proceed in two stages.

Firstly, for $j \in \{0, 1\}$, the targets are selected with

$$t_j^* \in \underset{t}{\operatorname{argmin}} \|\mathbf{I}(s) - \mathbf{I}(t)\|_2^2 - \eta \|\vec{s}t\| \leq \|\mathbf{g}_s, \vec{s}t\|, \quad (3.17)$$

where

$$t \in \left\{ \begin{array}{l} t \in \mathcal{S}, (1 - \epsilon^2) \leq \vec{s}t, \mathbf{g}_s \geq 2 + \left\| \vec{s}t - \frac{\vec{s}t, \mathbf{g}_s}{\|\mathbf{g}_s\|} \vec{s}t \right\|^2 < \frac{A_V}{\pi\sqrt{1-\epsilon}}, \\ \text{with } \langle \vec{s}t, \mathbf{g}_s \rangle (-1)^j > 0 \end{array} \right\}, \quad (3.18)$$

where $\eta \in \mathbb{R}_{>0}$ is an hyper parameter to be set, and $\|\cdot\|$ denotes the Euclidean norm. In Equation (3.17), the first term favors the sites s and t to have similar image intensities while the second term favors far targets that are aligned with \mathbf{g}_s . We note that the range of search in Equation (3.18) is limited to an ellipse that corresponds to the one of the shape-based neighborhood introduced in Equation (3.16).

Secondly, the paths are selected among the two sets $\Pi_\delta(s, t_j^*)$ joining s to t_j^* . For doing so $\forall j \in \{0, 1\}$, we formulate a cost function that the optimal path p_j^* has to minimize:

$$p_j^* \in \underset{p \in \Pi_\delta(s, t_j^*)}{\operatorname{argmin}} \sum_{k=0}^{|p|-1} \|\mathbf{I}(p(k)) - \mathbf{I}(p(k+1))\|_2^2, \quad (3.19)$$

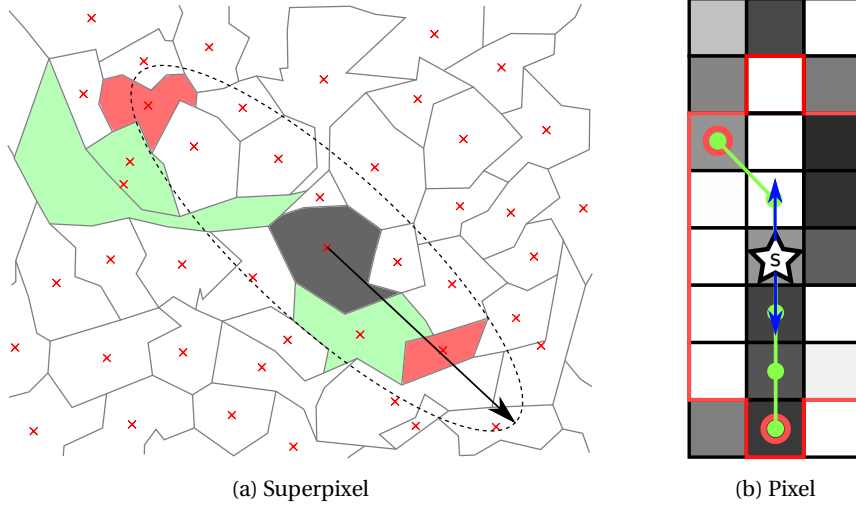


Figure 3.14 – Target-based neighborhood. Two “targets” (in red) are selected in an ellipsis centered on the source site s (in grey in (a), represented with a star in (b)). The arrow indicates the orientation of \mathbf{g}_s . For joining them, adjacent sites are selected (in green) until they form two paths towards the targets. We note that at pixel level (b) the target selected may not be perfectly aligned with \mathbf{g}_s , since small deviations are allowed to follow the structures of the image represented in shades of gray.

where $|p|$ stands for the length of the path, and $p(k)$ is the k^{th} element of the path. The term to be minimized in Equation (3.19) is large when image intensities of successive sites along a path are dissimilar and small otherwise. The neighborhood $V(s)$ can then be constructed as follows :

$$V(s) = (p_0^* \cup p_1^*) \setminus \{s\}.$$

We give an illustration of this neighborhood in Figure 3.14. While this neighborhood ensures that the set of neighbors of a site and s is a single connected component, and favors paths oriented in the direction of estimated thin structures, there is no guarantees concerning the cardinality of these paths. Depending on the image content that influences the location of the target sites, one site may have a very small amount of neighbors in a very contrasted location, or conversely could possibly have a high number of neighbors if there exists an arbitrary long path with constant radiometry. Coupled with the fact that the sites may be superpixels with a complex geometry, this could lead to unrealistic neighborhoods.

In practice however, constant radiometry in real life images is quite rare, and in our implementation, we overcome this problem by restricting the path search to sites whose barycenters lie within a large disc. However to evaluate the benefice of ensuring a constant number of neighbors for each site, we have also implemented a cardinal-based neighborhood.

Cardinal-based neighborhood

The idea of cardinal-based neighborhood is quite simple: Constructing, $\forall s \in \mathcal{S}$, $V(s)$ as the union of two cardinal-fixed paths $p_0^*, p_1^* \in \Pi_{\delta K}(s, \cdot)$ where the elements of $\Pi_{\delta K}(s, \cdot)$ are the paths of length $K \in \mathbb{N}_{>1}$ starting from s . Additionally, these paths are encouraged to expand in the two opposite directions derived from the guidance map \mathbf{g} . In this way, the cardinality of the neighborhood is fixed.

A cost function is set for allowing the paths to follow the structures of the image: As previously, there is a trade off between fidelity to the thin structure orientation and fidelity to the radiometry of the reference site. For any $j \in \{0, 1\}$, these paths are computed as:

$$p_j^* \in \operatorname{argmin}_{p \in \Pi_{\delta K}(s, \cdot)} l_C^j(p), \quad (3.20)$$

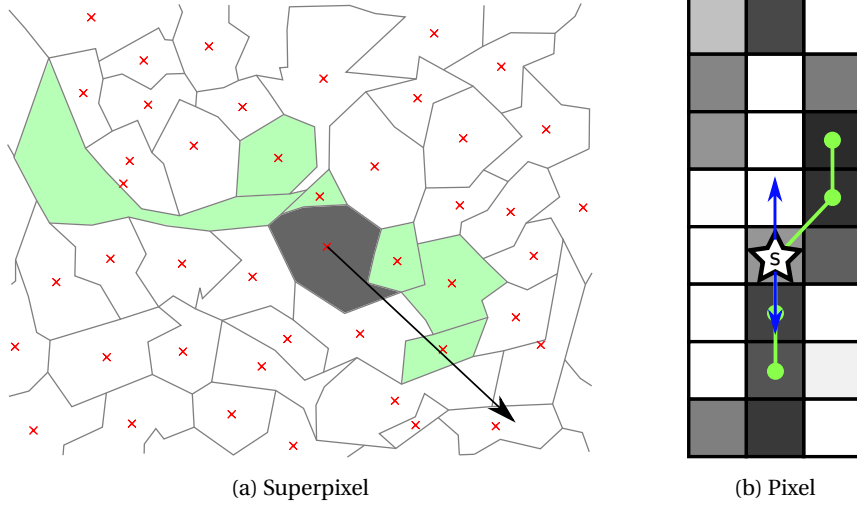


Figure 3.15 – At superpixel level (a), for a source site s in grey, the cardinal-based neighborhood builds two paths with the same number of elements K . Ideally, those paths are aligned with \mathbf{g}_s , whose direction is indicated by the arrow, but the energy term for selecting the paths also includes a radiometric similarity measure with the site s to allow deviations from this axis for following thin structures in the image. This fact is illustrated at pixel level (b): from the source site s , deviations of the neighborhood (in green) from the axis (blue arrows) are authorized in order to follow the structures represented by darker pixels.

with $l_C^j(p)$ the cost function associated with the path p , defined by :

$$l_C^j(p) = \sum_{t \in p} \|I(s) - I(t)\|_2^2 + \eta' \psi_j(\vec{s}t, \mathbf{g}_s), \quad (3.21)$$

where $\eta' \in \mathbb{R}_{>0}$ is an hyper parameter to set, and

$$\psi_j(\vec{u}, \vec{v}) = \begin{cases} \arccos(|\langle \vec{u}, \vec{v} \rangle|) & \text{if } (-1)^j \langle \vec{u}, \vec{v} \rangle > 0, \\ +\infty & \text{otherwise,} \end{cases}$$

measures the angle between the vectors \vec{u} and \vec{v} and discriminates whether the scalar product is positive or not.

The first term of Equation (3.21) encourages the image intensities of any site s_i to be similar to s intensity while the second term aims at aligning the path with \mathbf{g}_s . We note that the cost function compares radiometry and positions of each site of the path versus the site s instead of computing these differences on the adjacent sites on the path, to allow local deviations while ensuring global neighborhood orientation and color.

Finally, the neighborhood $V(s)$ of the site s can be now constructed as follows:

$$V(s) = (p_0^* \cup p_1^*) \setminus \{s\}$$

We give an illustration of this neighborhood in Figure 3.15. In anisotropic regions, this definition ensures the construction of a thin neighborhood with a constant cardinal while constraining the sets of neighbors and their origins to be single connected components.

3.4 Conclusion

In this chapter, we have presented the main contributions of this work.

We adapted TV, path opening and dilation operators at superpixel level, for performing thin structure detection on input images. We propose three alternatives. Firstly, an energy-based guidance map, that is computed from the energy of the neighborhood configurations, ensures the regularization of the neighborhood orientations with an energy term. Secondly, we propose to use TV

at superpixel level for extracting thin structures orientations, since it naturally provides smooth orientation maps that allow for edge continuation. Finally, we also introduced RORPO since it is tailored for thin structure detection, and uses path-based operators that offer consistency with the path-based neighborhoods that we introduce in the second half of this chapter.

The neighborhoods proposed split in two categories and are inspired by the two neighborhoods that we can compare to at superpixel level: The neighborhood of [Stawiaski and Decencière \[2011\]](#), and the disc-based neighborhood derived from [Giraud et al. \[2017a\]](#)'s *superpatch* formulation. The first one is computed from the notion of adjacency of sites only, while the second one is computed from the relative distances between the barycenters of the sites. The neighborhoods we propose base on these two aspects: The shape-based neighborhood introduces the notion of directivity in the analysis of the relative positioning of the sites, and can therefore be seen as a specialization of the disc-based neighborhood, while the path-based neighborhoods additionally integrate the notion of adjacency.

A third kind of neighborhood, the dictionary-based neighborhood, is closely related to the computation of energy-based guidance map, and can be seen as a constrained version of the other neighborhoods, even if, in our experimental study, we have only implemented its shape-based version.

In the next chapter, we explicit the way we minimize the energies using such neighborhoods, with graph cut, and we detail the implementation of our approach, including the tuning of the parameters introduced in this chapter.

Chapter 4

Numerical resolution

Contents

| | |
|--|-----------|
| 4.1 Optimization with graph cuts | 44 |
| 4.1.1 Definitions | 44 |
| Cuts in graphs | 44 |
| Flow in graphs | 46 |
| The max-flow/min-cut duality | 46 |
| 4.1.2 Energy minimization and max-flow algorithms | 47 |
| Graph representation | 47 |
| Graph construction | 47 |
| Max-flow algorithms | 49 |
| 4.1.3 From binary to multi-labels problem resolution | 53 |
| 4.2 Functional minimization | 59 |
| 4.2.1 Dictionary | 59 |
| 4.2.2 TV and RORPO based neighborhood | 60 |
| 4.2.3 Direct neighborhood estimation | 60 |
| 4.3 Implementation | 61 |
| 4.3.1 Superpixel generation | 61 |
| 4.3.2 Programming | 62 |
| Neighborhood precomputation | 62 |
| Graphical User Interface | 62 |
| Git repository | 65 |
| 4.3.3 Parameters setting | 65 |

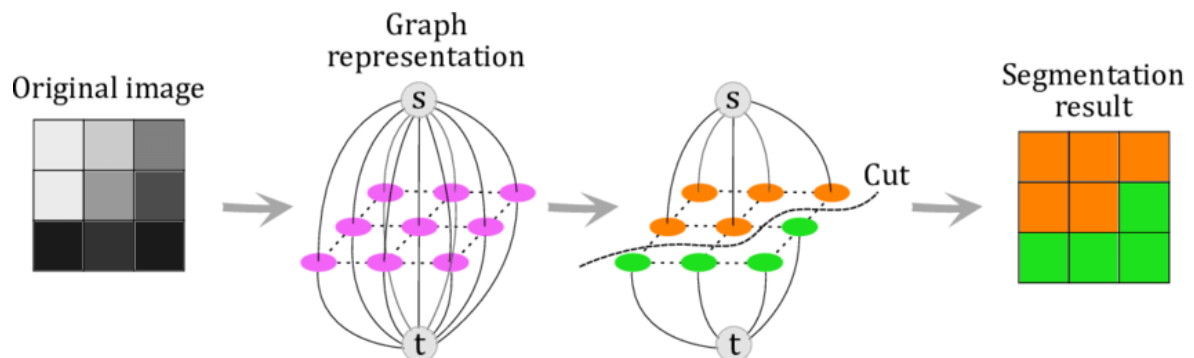


Illustration of graph cuts, Courtesy of Gauriau [2015].

In the previous chapters, we have presented some applications involving thin structures and demonstrated the need for anisotropic regularization. Then, we have introduced a new functional, discussed its formulation and defined new neighborhoods that can be considered as a second field to be estimated. The methodology for estimating and constructing these neighborhoods are detailed in Chapter 3.

In this chapter, we detail the numerical resolution of our problem, and its implementation. First, we introduce [Graph Cuts \(GC\)](#) in Section 4.1 and their use for efficiently minimizing functionals. Then, we present the actual algorithm we implement for minimizing our functional (Equation (2.3) or (3.7)), in Section 4.2. We present some details of our implementation in Section 4.3 and finally the setting of the parameters in Section 4.3.3.

4.1 Optimization with graph cuts

The problem of minimizing functionals is related to mathematical optimization. More specifically, when it comes to the labeling $\mathbf{u} \in \mathcal{C}^S$, it is a discrete problem with a finite number of elements, thus falling into the category of combinatorial optimization.

It is now necessary to discuss the actual method used for regularizing the blind segmentation with the functionals formulated in Equation (2.3) and Equation (3.7), depending on the application. Many methods coexist for minimizing a functional, such as [Iterated Conditional Modes \(ICM\)](#), [Belief Propagation \(BP\)](#), GC, or [Tree Reweighted Message Passing \(TRW\)](#). Those four methods have been compared on typical vision tasks in [Szeliski et al. \[2008\]](#) such as stereo imaging, image stitching, interactive segmentation and denoising.

For instance, ICM [\[Besag, 1986\]](#) is deterministic, greedy, iterative and converges rapidly: ICM starts with an initial estimation of the labeling, and iteratively scans each site, allowing a change of label only if the energy function is decreased. Without surprises, the result is extremely sensitive to the initialization and, in practice, of very poor quality.

BP [\[Felzenszwalb and Huttenlocher, 2006\]](#) iteratively minimizes the energy by transmitting messages between neighbors. Each message is computed from a data term and the previous messages received from neighbors. When a stopping criterion is reached, the label of each node is computed from these data. This method is however an approximation in the case of graphs integrating loops.

The conclusion of the comparative study of [Szeliski et al. \[2008\]](#) is that TRW and GC remain particularly competitive compared to the other algorithms, in terms of segmentation accuracy. For this reason, we decided to select GC for minimizing our functionals. Historically, due to limited resources and algorithm developments, GC remained limited to binary image restoration for a long time [D. Greig et al. \[1989\]](#). The emergence of fast maximum-flow/minimum-cut algorithms [Boykov and Kolmogorov \[2004\]](#) coupled to a better characterization of what energies can be minimized [Kolmogorov and Zabih \[2004\]](#), has been a milestone for solving challenging vision tasks such as segmentation, restoration, stereovision, etc. While [Kolmogorov and Zabih \[2004\]](#) prove their results with energies that write as a function of cliques of order less or equal to three, [Freedman and Drineas \[2005\]](#) generalize this for k -order cliques involving $k \in \mathbb{N}_{>0}$ sites.

In this Section, we first present the formalism of graphs, cuts and flow (Section 4.1.1), and then show how they are used in max-flow/min-cut algorithms (Section 4.1.2) for solving binary problems. Finally, we remind how multi-label problems can be solved by a decomposition into binary subproblems in Section 4.1.3.

4.1.1 Definitions

Cuts in graphs

Before describing the minimization algorithm, we present how the construction of a graph allow for modeling a functional, as reviewed in [Boykov et al. \[2001\]](#).

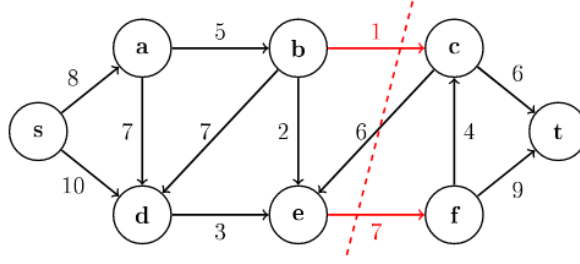


Figure 4.2 – An $s - t$ cut, represented as the red dashed line, with value 8. Note that the backward edges are not taken into account in the computation of the cost of the cut. Only forward edges, oriented from the source to the sink, are computed.

First, a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$ is a set of vertices \mathcal{V} connected by a set of oriented edges $\mathcal{E} \subset \mathcal{V}^2$:

$$\begin{cases} \mathcal{V} &= \mathcal{S} \cup \{s, t\} \\ \mathcal{E} &= \mathcal{N} \cup \mathcal{E}_T \\ c &: \mathcal{V}^2 \mapsto \mathbb{R}_{\geq 0}, \end{cases} \quad (4.1)$$

where the edge capacities are denoted c . The set of vertices includes two specific ones called terminals, s and t , called respectively the source and sink of the graph \mathcal{G} . Typically, in image segmentation, the other vertices of \mathcal{V} correspond to the set of sites \mathcal{S} , for instance a subset of \mathbb{Z}^2 with images of pixels. The edges linking the pairs of vertices, terminals excluded, are called n -links, and are typically the set of cliques \mathcal{N} in image segmentation. Finally, the set of edges $\mathcal{E}_T = (\{s\} \times \mathcal{S}) \cup (\{t\} \times \mathcal{S}) \subset \mathcal{E}$ that link the vertices to the terminals s and t are called t -links.

We recall that the set of cliques $\mathcal{N} \subset \mathcal{S}^2$ depends on the choice of the neighborhood. The capacities in the graph \mathcal{G} also respect the following properties:

$$\begin{aligned} \forall (p, q) \in \mathcal{E}, c(p, q) &> 0, \\ \forall (p, q) \notin \mathcal{E}, c(p, q) &= 0, \end{aligned}$$

A cut is the removal of a set of edges $\mathcal{C} \in \mathcal{E}$ from the graph \mathcal{G} . An s - t cut \mathcal{C} creates a partition of \mathcal{V} by removing edges of \mathcal{G} such that the terminals are separated in the induced graph $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s, c_s)$ and $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, c_t)$, with $s \in \mathcal{V}_s$ and $t \in \mathcal{V}_t$, see Fig 4.2. The partition means that $\mathcal{V}_s \cap \mathcal{V}_t = \emptyset$ and $\mathcal{V}_s \cup \mathcal{V}_t = \mathcal{V}$. These two graphs are defined as

$$\begin{aligned} \mathcal{G}_s &= (\mathcal{V}_s, \mathcal{E}_s, c_s), \text{ where } \begin{cases} \mathcal{V}_s, \\ \mathcal{E}_s &= \mathcal{E} \cap \mathcal{V}_s^2, \\ c_s &: \mathcal{V}_s^2 \mapsto \mathbb{R}_{\geq 0}, \end{cases} \\ \mathcal{G}_t &= (\mathcal{V}_t, \mathcal{E}_t, c_t), \text{ where } \begin{cases} \mathcal{V}_t, \\ \mathcal{E}_t &= \mathcal{E} \cap \mathcal{V}_t^2, \\ c_t &: \mathcal{V}_t^2 \mapsto \mathbb{R}_{\geq 0}. \end{cases} \end{aligned}$$

In the context of an s - t cut, a forward edge is an oriented edge $(p, q) \in \mathcal{E}$ such that $(p, q) \in (\mathcal{V}_s \times \mathcal{V}_t)$, while a backward edge is an edge $(q, p) \in (\mathcal{V}_t \times \mathcal{V}_s)$. What is called the cost, or the capacity, or the weight of an s - t cut \mathcal{C} , denoted $|\mathcal{C}|_{\mathcal{G}}$, is the sum of the capacities of the forward edges composing the cut. By concision, we omit \mathcal{G} and use the simplified notation $|\mathcal{C}|$ in the next parts of this paper. The cost of such an s - t cut can also be written as:

$$|\mathcal{C}| = \sum_{(p, q) \in (\mathcal{V}_s \times \mathcal{V}_t)} c(p, q) = \sum_{(p, q) \in \mathcal{C}} c(p, q). \quad (4.2)$$

Solving the minimum cut (or min-cut) problem amounts to finding the cut \mathcal{C}^* of minimum cost among all the s - t cuts. By definition, the min-cut is the cut that corresponds to the s - t cut with the minimum cost. In image segmentation, there is typically about $2^{\#\mathcal{S}}$ different s - t cuts, which makes any extensive search impractical. However, it is possible to find the min-cut in (pseudo-)polynomial time, thanks to the max-flow/min-cut duality.

We present this duality with the definition of the flow in the next subsection, while the way the min-cut is found is the subject of Section 4.1.2.

Flow in graphs

The flow is a quantity that respect some convenient properties, in particular: The flow is defined for each edge $e \in \mathcal{E}$ of the graph \mathcal{G} , where it has a non-negative value, the flow is bounded by above by the capacity of the edges, and finally, the flow is conservative in most the vertices. Some specific vertices, often the terminals may however deviate from the conservative flow rule, which explains the terms of *source* and *sink*. The flow pushed from any vertex is the sum of the flows in the edges that originate from this vertex, and for any vertex but the terminals, this sum is also equal to the flow pushed to this vertex, by conservative law.

In this section, to be in accordance with the notation of combinatorial optimization literature, we denote the flow by $f \in \mathbb{R}_{\geq 0}^{\mathcal{E}}$, where $f(p, q)$ is the flow in edge $(p, q) \in \mathcal{E}$ and its properties write as:

$$\begin{cases} \forall (p, q) \in \mathcal{E} & c(p, q) \geq f(p, q) \geq 0, \\ \forall p \in \mathcal{V} \setminus \{s, t\} & \sum_{(q,p) \in \mathcal{E}} f(q, p) - \sum_{(p,q) \in \mathcal{E}} f(p, q) = 0. \end{cases} \quad (4.3)$$

The value of the flow f in the graph \mathcal{G} is defined as follows:

$$|f| = \sum_{(s,p) \in \mathcal{E}} f(s, p). \quad (4.4)$$

It is the amount of flow injected in the graph from the terminal s . *Solving the maximum flow (or max-flow) problem amounts to finding the flow of maximum value and is the solution of the linear program:*

$$|f^*| = \begin{cases} \max_{f \in \mathbb{R}_{\geq 0}^{\mathcal{E}}} & |f|, \\ \text{such that} & f \text{ respects Equation (4.3).} \end{cases} \quad (4.5)$$

Loosely speaking, the max-flow is the maximum “amount of water” that can be sent from the source s to the sink t by interpreting graph edges as directed “pipes” with fixed capacities. An edge is *saturated* when the amount of flow that crosses it is equal to its capacity. The flow that is pushed from the source to the sink can be seen as an amount of water that we try to push through a set of pipes with limited capacity. A cut can be seen as a hyper-surface separating the source and the sink, and crossing the set of pipes at some points, from which we can measure a flow or compute the sum of the capacities of the intersected canals.

One of the fundamental results in combinatorial optimization is that the min s - t cut problem can be solved by finding a max-flow in a graph with two terminals the source s to the sink t .

The max-flow/min-cut duality

Because the flow is conservative, if the value of the flow that is pushed from the source to the sink is fixed, each cut has the same amount of flow through its surface. At the same time, the flow is bounded from above by the capacity of the cut. The increasing flow pushed from the source is therefore upper bounded by the cut that has the minimum capacity, that behave as a “bottleneck”.

The theorem in **Ford and Fulkerson [1956]** states that, for any oriented graph \mathcal{G} , any couple of vertex (s, t) , and any set of capacities c , the maximum value of the flow from s to t is equal to the smallest capacity of any s - t cut: $|f^*| = |\mathcal{C}^*|$.

\mathcal{C}^* can in addition be deduced from f^* : \mathcal{C}^* is the min-cut composed of a set of saturated edges that realize a s - t cut. Some edges may be saturated but not belong to the min-cut: A cut is minimum only if any edge removed from the cut would connect the two sets \mathcal{V}_s and \mathcal{V}_t . Therefore, edges belonging to \mathcal{C}^* are saturated edges but they may exist saturated edges that do not belong to \mathcal{C}^* . Because the flow function f is a function that is maximized, there also may exist multiple min-cuts for f .

The duality relationship between max-flow and min-cut problems motivates the need for an efficient algorithm for maximizing the flow in graphs. Fortunately, a wide range of algorithms exist for performing such operation, and therefore can be used for solving min-cut problems. Such algorithms are detailed in Section 4.1.2.

These algorithms allow for solving max-flow problems, but may also be used in the context of multi-source and multi-sink flow problems, and allow for computing the maximum number of disjoint paths from the source to the sink or solving problem of maximum cardinality matching in bipartite graphs.

4.1.2 Energy minimization and max-flow algorithms

Graphs allow for representing some functionals, and min-cut/max-flow algorithms allow for finding their minimizers. In this section, we present the energy functions that can be represented by graph cuts, as well as the corresponding construction of the graph and the max-flow algorithm used for solving binary and multi-label problem. While this allows for solving only binary problems, we describe later in Section 4.1.3 how these results can be extended in the multi label case.

Graph representation

The graph-representability of a function $E : \{0, 1\}^{\mathcal{S}} \rightarrow \mathbb{R}$ is defined as follows. Any function E is graph-representable if there exists a graph \mathcal{G} such that for any s - t cut \mathcal{C} ,

$$|\mathcal{C}| = E(\mathbf{u}^{\mathcal{C}}) + K, \quad (4.6)$$

where $K \in \mathbb{R}$ is a constant that has no impact on the minimizer and $\mathbf{u}^{\mathcal{C}}$ is the segmentation corresponding to the cut \mathcal{C} . When K is null, the energy is said to be *exactly* represented by \mathcal{G} .

Notice that there is a one-to-one correspondence between the possible configurations of the MRF, i.e. the labeling in $\{0, 1\}^{\mathcal{S}}$, and the $2^{|\mathcal{S}|}$ possible s - t cuts of the graph. When the energy is *graph-representable*, one is able to build a graph such that the cost of any s - t cut is equal to the energy of the configuration that corresponds to the s - t cut. For any s - t cut splitting the graph \mathcal{G} into two subgraphs \mathcal{G}_s and \mathcal{G}_t , and for any node $p \in \mathcal{V}$, the one-to-one correspondence is formalized by the following equation:

$$u_p^{\mathcal{C}} = \begin{cases} 0 & \text{if } p \in \mathcal{V}_s, \\ 1 & \text{if } p \in \mathcal{V}_t. \end{cases} \quad (4.7)$$

The graph-representability of energy functions is detailed in [Kolmogorov and Zabih \[2004\]](#) and [Freedman and Drineas \[2005\]](#) for a specific class of functions denoted \mathcal{F}^k where $k \in \mathbb{N}_{>0}$. Given a set of variables $\mathbf{u} = (u_p)_{p \in \mathcal{S}} \in \{0, 1\}^{\mathcal{S}}$, the class \mathcal{F}^k are functions $E : \{0, 1\}^{\mathcal{S}} \rightarrow \mathbb{R}$ that write as a sum of terms of up to k variables at a time. When $k = 2$, any function E belonging to \mathcal{F}^2 writes:

$$E(\mathbf{u}) = \sum_{p \in \mathcal{S}} E_p(u_p) + \sum_{(p,q) \in \mathcal{S}^2} E_{pq}(u_p, u_q). \quad (4.8)$$

The Theorem 3 in [Kolmogorov and Zabih \[2004\]](#) shows that any $E \in \mathcal{F}^2$ is graph-representable if and only if $\forall (p, q) \in \mathcal{N}$:

$$E_{pq}(0, 0) + E_{pq}(1, 1) \leq E_{pq}(1, 0) + E_{pq}(0, 1). \quad (4.9)$$

The condition (4.9) is called *submodularity*. A function that satisfies this condition is said to be submodular or regular. The representation of submodular energies is not unique, but [Kolmogorov and Zabih \[2004\]](#) propose a simple and compact graph construction for such energies, and show in their article that a global and exact minimizer of E can be found with a max-flow/min-cut algorithm.

Graph construction

We detail in this subsection the construction of \mathcal{G} for representing the energy E in Equation (4.8), for both the unary and the pairwise terms in which it can be decomposed. We remind that in this section the vertex s and t are the terminals of the graph \mathcal{G} , and that the proposed construction is not unique.

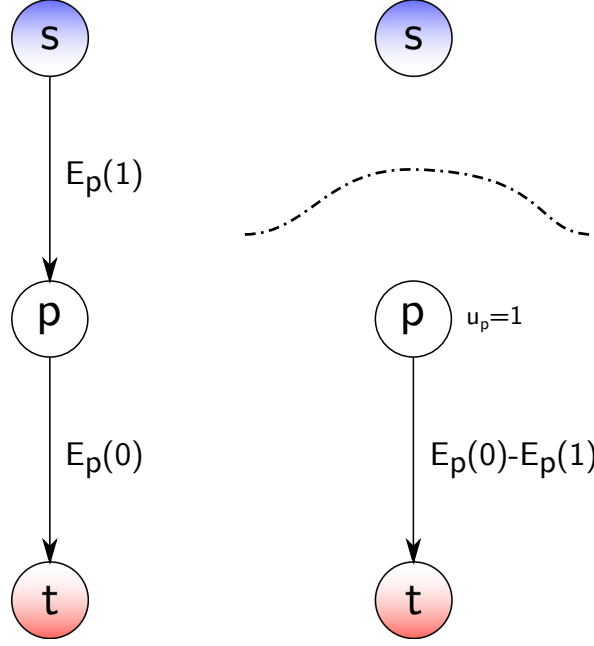


Figure 4.3 – Graph construction for unary terms of \mathcal{F}^1 . One of the two edges can be removed by subtracting the same quantity on the two edges capacities until one of them becomes null. Reducing the number of edges reduces the time and memory consumption of max-flow algorithms. In this example, the vertex p is labeled 1 by the cut \mathcal{C} , i.e. $u_p = 1$.

Unary terms: any term E_p is graph-representable: $\forall p \in \mathcal{S}$, we can set, without loss of generality:

$$\begin{cases} c(s, p) = E_p(1) - \min(E_p(0), E_p(1)), \\ c(p, t) = E_p(0) - \min(E_p(0), E_p(1)). \end{cases} \quad (4.10)$$

A s - t cut will necessarily cut either edge (s, p) or (p, t) , but not both since the cut has to be minimal. If the edge from the source to the vertex is cut, the energy is $E_p(1)$, and in this configuration the label u_p is set to 1. If the edge from the vertex to the sink is cut, the energy is $E_p(0)$, and the label u_p is set to 0 (see Figure 4.3).

However, the fact that the energy may be exactly represented or not does not impact the labeling associated with the minimum s - t cut. Thus, we may set $K \neq 0$ in Equation (4.6) and add or subtract any quantity to both edge capacities. Subtracting the minimum value $\min(E_p(0), E_p(1))$ to both edges capacities simplifies the graph by removing at least one of the edges, and thus reduces the memory footprint.

Pairwise terms: the decomposition of terms involving second order cliques E_{pq} that are sub-modular is a bit more complex, but can be understood by visualizing the s - t cuts corresponding to any configuration, see Figure 4.4. The capacities of the edges are non-negative values, therefore negative values in the decomposition correspond to backward edges. Explained differently, it corresponds to an increase of K in Equation (4.6) by this negative value so that every term is positive.

The energy is decomposed into three components. First, a term that depends on u_p , then a term that depends on u_q and finally a term that depends on the variable $(u_p) \cdot \overline{u_q}$. We have detailed it in Table 4.1.

In particular, when $E_{p,q}(0, 0) = E_{p,q}(1, 1) = 0$, the graph can be simplified with only two n -links. This is typically often the case for segmentation applications, since the Gibbs field that the MRF represents often deals with the attractive case when neighbors tend to share the same configurations. With most functionals in image segmentation, the global graph representing the energy E is therefore constructed so that t -links encode the energy terms E_p , and the n -links encode for the terms E_{pq} .

| | | | |
|----------------|----------|--------------------|--------------------------------------|
| $E_{00} + K =$ | E_{01} | $+E_{00} - E_{01}$ | $+0$ |
| $E_{01} + K =$ | E_{01} | $+0$ | $+0$ |
| $E_{11} + K =$ | E_{11} | $+0$ | $+0$ |
| $E_{10} + K =$ | E_{11} | $+E_{00} - E_{01}$ | $+E_{01} + E_{10} - E_{00} - E_{11}$ |

Table 4.1 – Decomposition of the energy $E_{pq} \in \mathcal{F}^2$ when $E_{00} > E_{01}$ and $K = 0$, from which other cases can be derived. For more readability in this table, we have written E_{00} for $E_{pq}(0, 0)$, E_{01} for $E_{pq}(0, 1)$, and so on. This decomposition is not unique, but allows for understanding the representation and the capacities of the edges in Figure 4.4. For keeping every capacity positive, any constant value may be added to any column of this table. For instance, we add $K = E_{01} - E_{00}$ to first and third column, so that the edge $c(q, t)$ with negative capacity is replaced by the edge $c(s, q)$ with positive capacity.

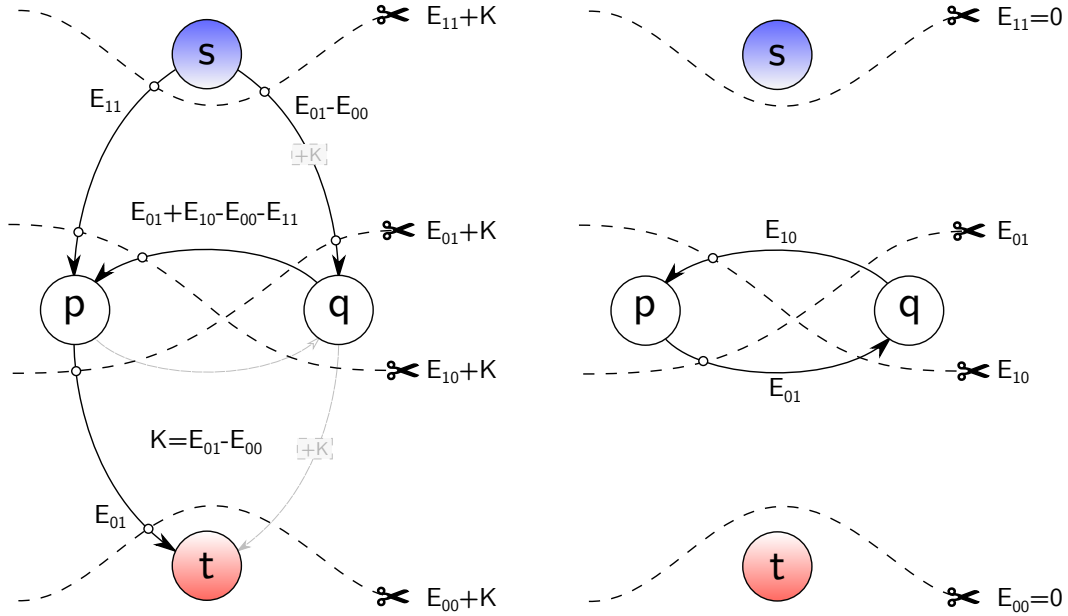


Figure 4.4 – Graph construction for energies of \mathcal{F}^2 . We have represented different cuts that illustrate the validity of the decomposition for the possible configurations. The graph on the left shows the positioning of the different edges when $E_{01} \geq E_{00}$, since it is often the case in image processing. We note that it corresponds to Table 4.1 with the value $K = E_{01}$ added to the third column. We recall that the cost of a s - t cut is the sum of the capacities of the edges cut from the source to the sink, while the backward edges are not taken into account in the cost of the cut. For convenience, we have written E_{00} for $E_{pq}(0, 0)$, E_{10} for $E_{pq}(1, 0)$, and so on. The graph on the right shows a simplification that can be written when $E_{00} = E_{11} = 0$ and setting $K = 0$.

The capacities of the edges of the global graph are computed as the sum of the capacities of the individual terms contribution.

With such construction allowing to represent the energy $E(\mathbf{u})$ of any configuration $\mathbf{u} \in \{0, 1\}^S$ as the capacity of a cut \mathcal{C} , the idea is now to compute the min-cut with an efficient max-flow algorithm.

Max-flow algorithms

There are multiple max-flow algorithms on graphs, used for solving the dual min-cut problem. In terms of complexity and because of the large number of vertices in the graph in typical computer vision tasks that often exceeds a billion for modest resolution cameras, it is important to use efficient algorithms.

As briefly introduced in Section 4.1.1, the idea is to push the maximum possible flow from the source to the sink, through the oriented edges of the graph. We recall that one property is that the flow is conservative, and bounded by above in each edge by the capacity of this edge. The intuitive idea of max-flow algorithm is to increase progressively the flow that emerges from the source to saturate edges, until the source and the sink of the graph are separated by a set of saturated edges

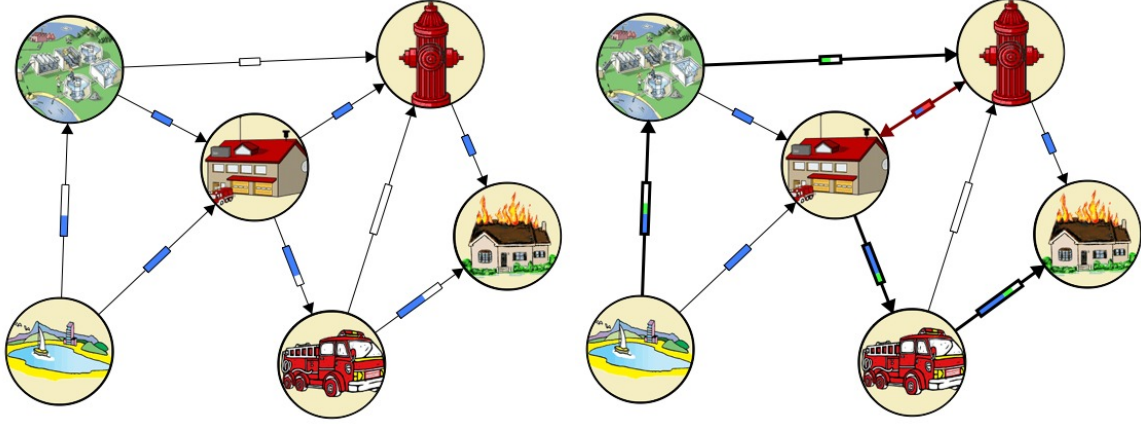


Figure 4.5 – Representation of the max-flow problem by analogy with the flow of water through a network of pipes. The idea is to bring the maximum amount (flow) of water from the source (the lake) to the sink (the house in fire). For doing so, the operator has multiple choices and multiple canals represented as the vertices and edges of the graph. The first image on the left presents a situation to which one may arrive by intuitively and in an iterative way pushing the maximum flow through each path that reaches the fire. However, it is still possible to find an augmenting path (thick line in the image on the right) that makes use of an edge in reverse order. This is an illustration of the approach proposed by Algorithm 1, and the min-cut in the graph is found among the saturated edges. (Image courtesy of Caruso and Fourquaux, CNRS and the Institut Henri Poincaré).

(the min-cut) that disconnect the graph. This is the idea of the Ford and Fulkerson algorithm that we introduce first. Next, we present some of the max-flow algorithms reviewed or introduced in Boykov and Kolmogorov [2004].

Ford Fulkerson max-flow algorithm: The max-flow algorithm in Figure 4.5 bases on the Ford Fulkerson iterative algorithm. The latter one progressively increases the flow pushed in a graph \mathcal{G} by finding *augmenting paths* in a *residual graph* \mathcal{G}_f .

The residual graph $\mathcal{G}_f = (\mathcal{V}, \mathcal{E}, c_f)$ shows the same vertices and edges as \mathcal{G} with edge capacities c_f reduced by the amount of flow through each edge. At the initial step, the flow is null, therefore the residual capacities of the edges of \mathcal{G}_f equal the capacities of the edges in \mathcal{G} :

$$c_f(e) = c(e), \quad \forall e \in \mathcal{E}. \quad (4.11)$$

It is a way of updating the available capacities of the graph instead of computing the differences between the flow and the capacities for each edge in the graph.

An augmenting path $\mathcal{P} = (e_0, \dots, e_k)$ with $k \in \mathbb{N}_{>0}$ is an ordered set of non saturated edges $e_i = (p_i, q_i) \in \mathcal{E}, \forall i \in \llbracket 0, k \rrbracket$, from the source to the sink that allow for pushing a flow through a graph such that:

$$\begin{cases} p_0 = s, \\ q_i = p_{i+1}, & \forall i \in \llbracket 0, k \rrbracket, \\ q_k = t, \\ c_f(p, q) > 0. \end{cases} \quad (4.12)$$

The maximum value δf of the flow through the augmenting path is reached when one of its edges becomes saturated. Indeed at this precise moment, the path is not an augmenting path anymore. Any backward edge \mathcal{E}_R can also be part of the augmenting path, as long as the total flow through each edge is non-negative. In this case, the total flow in backward edges is reduced by the additional amount of flow δf . δf is the minimum value between the minimum residual capacities of forward edges and the minimal flow of the backward edges. For any forward edge (p, q) joining two sites $(p, q) \in \mathcal{S}^2$, the associated backward edge that links q to p is denoted by (p, q) .

The **Ford and Fulkerson (FF)** algorithm is described below:

- Initialize \mathcal{G}_f from \mathcal{G} .
- Repeat the following steps while an augmenting path is found:
 1. Find an augmenting path \mathcal{P} from s to t in \mathcal{G}_f
 2. Compute $\delta f = \min_{(p,q) \in \mathcal{P}} c_f(p, q)$
 3. Update the capacities of the edges: $\forall (p, q) \in \mathcal{P}$,
 $c_f(p, q) \leftarrow c_f(p, q) - \delta f$
 $c_f(q, p) \leftarrow c_f(q, p) + \delta f$
 4. Update the total flow $f \leftarrow f + \delta f$.
- Find the min s - t cut among the saturated edges of \mathcal{G}_f , which value is f .

Algorithm 1: Max-flow algorithm of FF.

At each iteration, the algorithm finds an augmenting path along non-saturated edges of the residual graph \mathcal{G}_f , and *augments* it by pushing into it the maximum possible flow δf . The residual capacities of edges in the path are reduced by δf , the residual capacities of backward edges are increased by δf while the total flow is increased by δf . Then, the maximal flow is reached when there is no more augmenting path from s to t .

The efficient search of a path from the source to the sink is a critical factor for processing the solution in a reasonable amount of time. In FF algorithm, the way augmenting paths are selected is not specified. Therefore, if the capacities are integers, in the worst case scenario the flow is only increased by one unit at a time, so the worst case complexity of the algorithm is $O(\#\mathcal{E}|f^*|)$ ($\#\mathcal{E}$ being the number of edges in the graph) which is pseudo-polynomial. Theoretically with real capacities, the algorithm may not even converge in very specific cases.

Some algorithms focus on paths with largest capacities (see [Edmonds \[1972\]](#)), leading to worst case time complexity $O(\#\mathcal{E}^2\#\mathcal{V})$, where $\#\mathcal{V}$ is the number of nodes. Another type of approaches that we describe in the following paragraph focus on finding the shortest paths, and sometimes even mix both approaches ([Juan and Boykov \[2007\]](#)).

[Dinic \[1970\]](#) uses breadth-first search to find the shortest paths from s to t in the residual graph \mathcal{G}_f . By opposition to a depth-first search algorithm that searches through the successors of a node until the end of a path, a breadth-first search is the exploration of a graph from a source node, and through all its successors, then all the successors of its successors, and so on. In such a search algorithm, the paths are explored in increasing length: After all shortest paths of a fixed length k are saturated, the algorithm starts the search for paths of length $k + 1$. The use of shortest paths improves the theoretical running time complexity for algorithms based on augmenting paths. The worst case running time complexity becomes $O(\#\mathcal{E}\#\mathcal{V}^2)$.

Boykov Kolmogorov max-flow algorithm In the context of graphs in computer vision, building a breadth-first search tree is a very expensive operation, since it involves scanning the majority of image pixels. [Boykov and Kolmogorov \[2004\]](#) based their algorithm on augmenting path techniques, but overcome the latter disagreement by building two search trees, one from the source and one from the sink, that are reused and never rebuilt from scratch. Reusing both the trees has a theoretical drawback: The complexity of their algorithm $O(\#\mathcal{E}\#\mathcal{V}^2|\mathcal{C}|)$ is greater than the one of shortest augmenting path since the augmenting path found from the updated trees are not necessarily the shortest ones. However, in practice in image processing, a linear complexity in the number of nodes is observed in the paper.

The algorithm of [Boykov and Kolmogorov \[2004\]](#) grows two non-overlapping trees $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s, c_s)$ and $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, c_t)$ with roots at the source s and the sink t . We illustrate this approach on Figure 4.6. Initially, these trees are reduced to $\mathcal{G}_s = (\{s\}, \emptyset, c_s)$ and $\mathcal{G}_t = (\{t\}, \emptyset, c_t)$. The algorithm is iterative and repeats the following three stages until convergence:

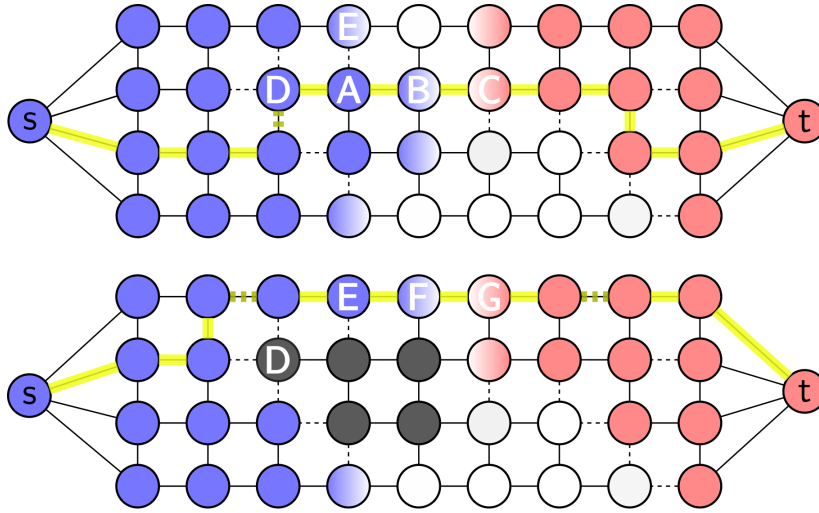


Figure 4.6 – The Boykov max-flow algorithm: Two search trees are built, starting from the source and the sink, respectively colored in blue and red. Active nodes are represented with a color gradient, while the inactive ones are uniformly colored. Orphaned nodes are shown in dark grey and free nodes in white. Available edges are represented as continuous lines, while saturated edges are represented as dotted lines. Both the trees are iteratively updated through the steps of growing, augmenting, and adoption. In this example, a node (B), active and child of (A), encounters a node (C) belonging to the opposite tree: the augmenting stage begins. The augmenting path is represented as a thick yellow line, whose smallest capacity edges are represented by darker yellow dotted thick edges. After the augmentation stage, the node (D) becomes an orphan for whom the adoption fails, therefore all the black orphan nodes in the second image recursively become free nodes. At the next growing stage, the acquisition by the node (E) of a child (F) exhibit a new augmenting path through the node (G).

- First, a growth stage expand trees \mathcal{G}_s and \mathcal{G}_t until they touch, thus giving a path. If no valid path is found, the algorithm has converged.
- Secondly, an augmentation stage pushes flow through the path, saturates some edges and generate orphaned nodes that break the search trees into forests.
- And lastly, the trees of the forests are gathered by reconnecting the orphaned nodes to one of the trees \mathcal{G}_s and \mathcal{G}_t in the adoption stage.

During the *growth* stage, the search trees expand. Each active node acquires new children from the set of free nodes by exploring adjacent non-saturated edges. These newly acquired nodes become active members of the corresponding search trees. As soon as all the neighbors of an active node are explored, it is relabeled as passive. As soon as an active node encounters a neighbor that belongs to the opposite tree, we assume a path is detected from the source to the sink, and thus, the growth stage terminates.

The *augmentation* stage augments the path found at the growth stage. The largest flow possible is pushed through the path and some edges in the path become saturated. Thus, the search trees \mathcal{G}_s and \mathcal{G}_t may split into forests and some nodes become orphans.

The goal of the *adoption* stage is to restore the tree structure of \mathcal{G}_s and \mathcal{G}_t with roots at the source and the sink. Each orphan node seeks for a new valid parent. A node can be a valid parent only if it belongs to the same tree as the orphan, and can be connected to it with a non saturated edge. If no valid parent is found, the orphan becomes a free node and all its former children become free nodes. Once all the orphans are explored and marked as active, passive or free nodes, the structure of the search trees \mathcal{G}_s and \mathcal{G}_t are restored, and the adoption stage terminates. Since orphans may become free nodes, we note that both trees may shrink during this step of the algorithm.

The algorithm then returns to the growth stage, until no augmenting path can be found, which means that the search trees are separated by saturated edges. This implies that the maximum flow

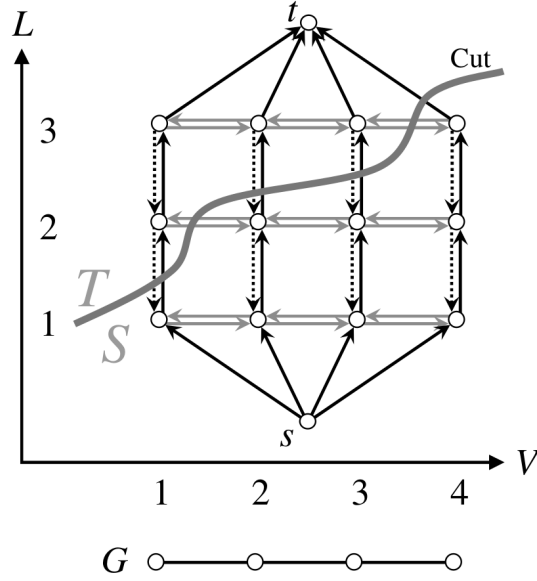


Figure 4.7 – An example of cut in a graph in a problem with three labels and four sites, where $E_{p,q}$ is a function of absolute difference between labels. Courtesy of Ishikawa [2003]. The black arrows represent data edges, while horizontal edges capacities are linked to the energy brought by the smoothness prior. Ishikawa [2003] also adds constraint edges of infinite capacity depicted as dotted arrows that prevent the cut from “going backward”. In this example, $v_1 = 1$, $v_2 = 2$, $v_3 = 2$, and $v_4 = 3$. Beyond absolute difference, any convex prior may be used at the expense of embedding more edges.

is reached. The min-cut and the final binary labeling is then recovered from the search trees which terminates the algorithm of Boykov and Kolmogorov [2004], $\forall p \in \mathcal{S}$:

$$\begin{aligned} u_p &= 1 \text{ if } p \in \mathcal{V}_s, \\ u_p &= 0 \text{ if } p \in \mathcal{V}_t. \end{aligned} \quad (4.13)$$

We used the implementation provided by Veksler and Delong¹, using double-precision floating-point format.

4.1.3 From binary to multi-labels problem resolution

Now that we have presented the way some binary energies can be exactly minimized by graph cuts, let us introduce how problems of minimization involving more than two labels $\mathbf{u} \in \mathcal{C}^{\mathcal{S}}$ may be minimized, when the energy is in the form:

$$E(\mathbf{u}) = \sum_{p \in \mathcal{S}} E_p(u_p) + \sum_{(p,q) \in \mathcal{N}} E_{pq}(u_p, u_q). \quad (4.14)$$

Multiple strategies exist for minimizing the above energy. But depending on the application, the user's choices depend both on the properties of the energy and the constraints of computation time and memory consumption. For instance, in the case of convex priors (pairwise terms) $E_{p,q}$ but with any unary terms E_p , Ishikawa [2003] proposes the construction of a multi-layer graph that also allows for an exact minimization of this energy (see Figure 4.7). While such an approach can *exactly* minimize non-convex energy functions, this construction is however very memory consuming, which makes it unsuitable for typical computer vision tasks, in which the order of magnitude of the number of nodes is around multiple billions of nodes. Such graphs would not fit into most computer's memory, and the computation of max-flow algorithm would be far beyond reach.

¹Currently available here <https://vision.cs.uwaterloo.ca/code/>.

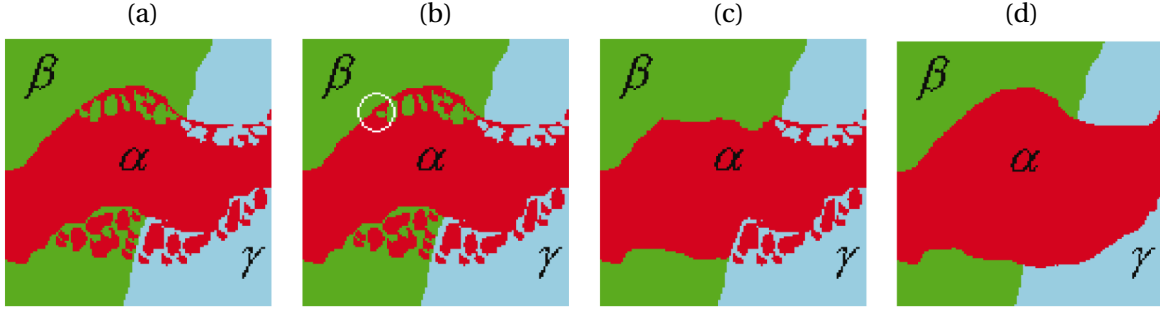


Figure 4.8 – Examples of labeling (a) evolving with swap (c) and expansion (d) moves compared to standard moves (b). While standard move only allow the change of the label of one site, α - β swap allow any site labeled α to be relabeled β , and reciprocally. α -expansion is an even more efficient move, where all the sites are free to be relabeled α or keep their previous labeling all at once (Picture courtesy of [Boykov et al. \[2001\]](#)).

In the general case, finding the optimal labeling for such energies is an NP-complete problem, which encourages the use of heuristic approaches, efficient if possible.

Instead of building very large graphs for solving the problem all at once, [Boykov et al. \[2001\]](#) have developed efficient algorithms for computing approximate minimizer of multi-label problems. To do so, they decompose any multi-label problem into a sequence of binary subproblems that are solved with binary graph cut (see Section 4.1.2). In their article, they present α - β swap and α -expansion, two heuristics. In α - β swap, each site labeled α or β is allowed to change its label to α or β , while the other sites remain unchanged. During α -expansion, every site is either assigned with the label α , or remains unchanged. Both heuristics allow for solving more efficiently the problem, but the drawback is that the labeling obtained after convergence is a local minimum, within a known factor of the global minimum. Both heuristics require the energy to respect some properties, that are stricter for α -expansion, that is a stronger move in the sense that it allows for a more faster resolution of the problem with better minimum guarantees.

Visually, the effect of a standard move (the change of the label of a single node in \mathcal{S}) and of stronger moves (expansion and swap moves) are pictured in Figure 4.8.

While both methods work well with ordered or unordered labels (respectively image restoration or image segmentation for instance), others approaches may work more efficiently when the level sets of the image are nested. [Darbon and Sigelle \[2006\]](#) use a divide-and-conquer approach: They decompose the problem into binary subproblems, solve independently each subproblem and recombine the solution in order to get the solution of the global problem. The same idea has for example been instantiated on SFF in our article [Ribal et al. \[2018\]](#). We detail the latter one a bit more, to illustrate our dyadic divide-and-conquer strategy.

α -expansion It is an iterative algorithm minimizing a sequence of submodular energies E' . Those energies are defined as subproblems that only involve binary variables and are classically solved with max-flow/min-cut algorithms. For each iteration, there are $\#\mathcal{C}$ subproblems since for each subproblem, a label is selected until the expansion moves $\forall \alpha \in \mathcal{C}$ are processed.

For each subproblem, $\alpha \in \mathcal{C}$ is fixed, and a second field $\mathbf{v} \in \{0, 1\}^{\mathcal{S}}$ determines for each node $p \in \mathcal{V}$ if its label remains unchanged or is assigned to the label α (this is the *expansion move*). The labeling \mathbf{u}^v computed from the resolution of the subproblem is defined as:

$$u_p^v = \begin{cases} u_p & \text{if } v_p = 0, \\ \alpha & \text{if } v_p = 1, \end{cases} \quad \forall p \in \mathcal{S}.$$

When the label of node p is set to 1, it corresponds to setting u_p to the label α in the global problem, and when the label of node p is set to 0, the label u_p remain unchanged. Thus, the subproblem only requires to compute the set of sites $p \in \mathcal{S}_{\bar{\alpha}} = \{p \in \mathcal{V} \setminus \{s, t\}, \text{ such that } u_p \neq \alpha\}$.

By construction, the energy E is guaranteed to decrease at each iteration with the minimization

of E' that is defined as:

$$E(\mathbf{u}') = E'(v) = \sum_{p \in \mathcal{S}_{\bar{\alpha}}} E'_p(v_p) + \sum_{(p,q) \in \mathcal{N}} E'_{pq}(v_p, v_q),$$

where

$$\begin{cases} E'_p(0) &= E_p(u_p), \\ E'_p(1) &= E_p(\alpha), \end{cases} \quad \text{and} \quad \begin{pmatrix} E'_{pq}(0,0) & E'_{pq}(0,1) \\ E'_{pq}(1,0) & E'_{pq}(1,1) \end{pmatrix} = \begin{pmatrix} E_{pq}(u_p, u_q) & E_{pq}(u_p, \alpha) \\ E_{pq}(\alpha, u_q) & E_{pq}(\alpha, \alpha) \end{pmatrix}.$$

The minimization of E' is done using a max-flow/min-cut algorithm. The idea is to perform expansion moves for every label $\alpha \in \mathcal{C}$, until convergence, as summarized in Algorithm 2.

INPUT: Elements permitting the computation of E

OUTPUT: An approximate minimizer of E

```

1: Initialize  $\mathbf{u} = (u_p)_{p \in \mathcal{S}}$ 
2: repeat
3:    $\mathbf{u}' = \mathbf{u}$ 
4:   for  $\alpha \in \mathcal{C}$  do
5:      $v \in \underset{v \in \{0,1\}^{\mathcal{S}}}{\operatorname{argmin}} E'(v)$ 
6:     for  $p \in \mathcal{S}_{\bar{\alpha}}$  do
7:       if  $v_p = 1$  then  $u_p \leftarrow \alpha$ 
8:     end for
9:   end for
10: until  $E(\mathbf{u}') \leq E(\mathbf{u})$ 
    
```

Algorithm 2: α -expansion algorithm.

α -expansion can only be applied when each subproblem involves an energy E' that is submodular (see Equation (4.9)), i.e. when:

$$E_{pq}(u_p, u_q) + E_{pq}(\alpha, \alpha) \leq E_{pq}(u_p, \alpha) + E_{pq}(\alpha, u_q), \quad \forall u_p, u_q, \alpha \in \mathcal{C}. \quad (4.15)$$

In most cases, $E_{pq}(\alpha, \alpha) = 0$, so the previous equation becomes a triangular inequality.

There are several strategies from this point that allow for a slight tuning of the algorithm and may impact the results, with the initialization of \mathbf{u} , the order in which the label are explored, and finally the stopping criterion.

The initialization strategy of \mathbf{u} impacts the minimization of E , because α -expansion may not converge to the global optimum solution and may fall into a local minimum. Winner-takes-all, random and constant label strategies are examples of initialization. Winner-takes-all strategy is to initialize \mathbf{u} with the labeling that minimize the energy E without regards to pairwise terms E_{pq} :

$$u_p = \underset{l \in \mathcal{C}}{\operatorname{argmin}} E_p(l), \quad \forall p \in \mathcal{S}. \quad (4.16)$$

Random initialization associates a label chosen uniformly at random in \mathcal{C} to each site. Constant label strategy imposes the same label $\alpha \in \mathcal{C}$ for all sites. In practice, α -expansion appears however to be robust against the way u is initialized.

For the same reasons of convergence, the order in which the labels are scanned may impact the final result. Strategies include scanning the labels forward or backward, or even randomly. Finally, the theoretical convergence speed of the algorithm is unknown. The user may thus add some arbitrary stopping criterion for speeding up the convergence. In practice though, the α -expansion typically converges in only a few iterations on typical vision tasks, and this parameter tuning is of minor importance.

While knowing that the α -expansion converges to a local minimum of the energy $E(\hat{\mathbf{u}})$ that is a poor guarantee in terms of energy minimization, [Boykov et al. \[2001\]](#) proved that the local minimum is within a known factor $2\gamma \in [1, \infty[$ of the global minimum of the energy $E(\mathbf{u}^*)$:

$$E(\hat{\mathbf{u}}) < 2\gamma E(\mathbf{u}^*),$$

where:

$$\gamma = \frac{\max_{(p,q) \in \mathcal{E}, \alpha \neq \beta \in \mathcal{C}} E_{pq}(\alpha, \beta)}{\min_{(p,q) \in \mathcal{E}, \alpha \neq \beta \in \mathcal{C}} E_{pq}(\alpha, \beta)}.$$

We refer to the [Boykov et al. \[2001\]](#) for the mathematical proof.

α - β swap The idea of α - β swap sketched in Figure 4.8 is quite similar to the one of α -expansion: It is an iterative algorithm subdividing the problem of minimizing Equation (4.14) into a set of subproblems involving binary variables only. Each subproblem deals with the minimization, in our case via graph cuts, of an energy E' that is submodular. A subproblem corresponds to a pair of labels of the global problem, therefore leading to $\#\mathcal{C}^2$ subproblems for each iteration.

For each subproblem, a pair of labels $(\alpha, \beta) \in \mathcal{S}^2$ is fixed. Only sites labeled α or β are allowed to change their label to be either α or β , that is why it is called a *swap move*. After the resolution of this subproblem, the field $\mathbf{v} \in \{0, 1\}^{\mathcal{S}}$ determines if any site $p \in \mathcal{V}$ previously labeled α or β may be relabeled differently with the following rule:

$$u_p^v = \begin{cases} u_p & \text{if } u_p \notin \{\alpha, \beta\}, \\ \beta & \text{if } u_p \in \{\alpha, \beta\} \text{ and } v_p = 0, \\ \alpha & \text{if } u_p \in \{\alpha, \beta\} \text{ and } v_p = 1, \end{cases} \quad \forall p \in \mathcal{S}$$

One important thing to note is that the graph built $\mathcal{G}_{\alpha\beta} = (\mathcal{V}_{\alpha\beta}, \mathcal{E}_{\alpha\beta}, c)$ may have a small number of vertices as it is only composed of the vertices labeled α or β together with the source and the sink s, t (that are sometimes renamed α and β in the literature). More precisely, $\mathcal{V}_{\alpha\beta} = \mathcal{S}_{\alpha\beta} \cup \{s, t\}$ where $\mathcal{S}_{\alpha\beta} = \{p \in \mathcal{S} / u_p \in \{\alpha, \beta\}\}$ and $\mathcal{E}_{\alpha\beta} \subset \{\mathcal{V}_{\alpha\beta}\}^2$. We also denote a subset of the cliques \mathcal{N} by $\mathcal{N}_{\alpha\beta} = \mathcal{N} \cap \{\mathcal{V}_{\alpha\beta}\}^2$, that is also a subset of the edges $\mathcal{E}_{\alpha\beta}$. [Boykov et al. \[2001\]](#) give a definition of the s - t cut that is equivalent to the formulation of an energy E' as follows:

$$E(\mathbf{u}^v) = E'(v) = \sum_{p \in \mathcal{S}_{\alpha\beta}} E'_p(v_p) + \sum_{(p,q) \in \mathcal{N}_{\alpha\beta}} E'_{pq}(v_p, v_q),$$

with, $\forall p \in \mathcal{S}_{\alpha\beta}$,

$$\begin{cases} E'_p(0) = E_p(\beta) + \sum_{\substack{(p,q) \in \mathcal{S} \\ (q) \notin \mathcal{S}_{\alpha\beta}}} E_{pq}(\beta, u_q), \\ E'_p(1) = E_p(\alpha) + \sum_{\substack{(p,q) \in \mathcal{S} \\ (q) \notin \mathcal{S}_{\alpha\beta}}} E_{pq}(\alpha, u_q), \end{cases}$$

and

$$\begin{pmatrix} E'_{pq}(0,0) & E'_{pq}(0,1) \\ E'_{pq}(1,0) & E'_{pq}(1,1) \end{pmatrix} = \begin{pmatrix} E_{pq}(\beta, \beta) & E_{pq}(\beta, \alpha) \\ E_{pq}(\alpha, \beta) & E_{pq}(\alpha, \alpha) \end{pmatrix}, \quad \forall (p, q) \in \mathcal{N}_{\alpha\beta}.$$

The algorithm is then quite similar to α -expansion, see Algorithm 3.

The condition of submodularity of each subproblem reduces in α - β swap to the following inequality, $\forall \alpha, \beta \in \mathcal{C}, \forall p, q \in \mathcal{S}_{\alpha\beta}$:

$$E_{pq}(\beta, \beta) + E_{pq}(\alpha, \alpha) \leq E_{pq}(\beta, \alpha) + E_{pq}(\alpha, \beta). \quad (4.17)$$

It is a much weaker requirement than Equation (4.15), and that is one of the main advantages of α - β swap over α -expansion at the expense of a greater algorithm complexity.

We can formulate almost the same remarks than for α -expansion: Convergence to a global minimum is not guaranteed. Several strategies of initialization of \mathbf{u} may impact the results, as well

INPUT: Elements permitting the computation of E

OUTPUT: An approximate minimizer of E

```

1: Initialize u
2: repeat
3:   u' = u
4:   for  $(\alpha, \beta) \in \mathcal{C}^2$  do
5:      $\nu \in \underset{\nu \in \{0,1\}^{\mathcal{I}_{\alpha\beta}}}{\operatorname{argmin}} E'(\nu)$ 
6:     for  $p \in \mathcal{S}_{\alpha,\beta}$  do
7:       if  $\nu_p = 0$  then  $u_p \leftarrow \beta$ 
8:       if  $\nu_p = 1$  then  $u_p \leftarrow \alpha$ 
9:     end for
10:  end for
11: until  $E(\mathbf{u}') \leq E(\mathbf{u})$ 
    
```

Algorithm 3: α - β swap algorithm.

as the order in which the label are scanned. For both topics, we advise the reader to refer to the paragraph about α -expansion.

In terms of control of the quality of the solution however, α - β swap gives no guarantees about the final energy of the local minimum, that can be arbitrary far away from the energy of the global minimum. Instead, [Boykov et al. \[2001\]](#) propose to initialize \mathbf{u} for α - β swap with the result of α -expansion used on an approached energy function. This approximated energy replaces the pair-wise energy terms by a Potts model, and allows for obtaining better results, both theoretically and empirically, but we will not detail further these specific cases of use.

Dyadic search The dyadic search takes advantage of a divide-and-conquer process to drastically reduce the number of minimum cuts required to minimize Equation (4.14) to $\log_2(\#\mathcal{C})$. It is possible to do so when the label set \mathcal{C} is ordered, and if the binary solutions are “nested”. In the sequel of this paragraph, we assume that the $M \in \mathbb{N}_{>0}$ labels $\{l_0, \dots, l_{M-1}\} = \mathcal{C}$ are ordered with respect to their indices in $\mathcal{M} = \{0, \dots, M-1\}$: $\forall M > i > j \geq 0, l_i > l_j$.

As explained in [Darbon and Sigelle \[2006\]](#), both the unary and pairwise terms of the functional can be decomposed as a sum of energies on the level sets of $\mathbf{u} \in \mathcal{C}^{\mathcal{S}}$. Let us denote by u_p^l the value at pixel $p \in \mathcal{S}$ of $\mathbf{u}^l = \left(\mathbf{1}_{\{u_p \geq l\}} \right)_{p \in \mathcal{S}} \in \{0, 1\}^{\mathcal{S}}$ the l -level set of the image \mathbf{u} . For any pixel $p \in \mathcal{S}$, the unary terms can be decomposed as a telescopic sum:

$$E_p(u_p) = \left(\sum_{0 < i < M} u_p^{l_i} (E_p(l_i) - E_p(l_{i-1})) \right) + E_p(l_0). \quad (4.18)$$

Note that the latter equation is consistent whatever $u_p \in \mathcal{C}$. Similarly, we decompose for any pixel pair $(p, q) \in \mathcal{N}$ the term:

$$E_{pq}(u_p, u_q) = \sum_{0 < i < M} E_{pq}(u_p^{l_i}, u_q^{l_i}). \quad (4.19)$$

Since $u_p^{l_0} = u_q^{l_0} = 1, \forall p, q \in \mathcal{S}$, the sum in the latter equation starts from $i = 1$.

The functional to minimize Equation (4.14) may now be written, using Equation (4.18) and Equation (4.19), $\forall (p, q) \in \mathcal{N}$:

$$E(\mathbf{u}) = \sum_{0 < i < M} E^{l_i}(\mathbf{u}^{l_i}) + K, \quad (4.20)$$

where $K \in \mathbb{R}$ is a constant that does not depend on \mathbf{u} , and the energies E^{l_i} are defined, for any $0 < i < M$ and any level set (or binary image) $\mathbf{u}^{l_i} \in \{0, 1\}^{\mathcal{S}}$, by

$$E^{l_i}(\mathbf{u}^{l_i}) = \sum_{p \in \mathcal{S}} u_p^{l_i} (E_p(l_i) - E_p(l_{i-1})) + \sum_{(p,q) \in \mathcal{N}} E_{pq}(u_p^{l_i}, u_q^{l_i}). \quad (4.21)$$

For any $k, k' \in \mathcal{M} \setminus \{0\}$, let us denote by $\mathbf{u}^{*l_k}, \mathbf{u}^{*l_{k'}} \in \{0, 1\}^S$ the minimizers of E^{l_k} and $E^{l_{k'}}$, respectively. If these minimizers satisfy

$$u_p^{l_k} \geq u_p^{l_{k'}}, \forall 0 \leq k \leq k' \leq M-1, \forall p \in S, \quad (4.22)$$

i.e. the level sets \mathbf{u}^{*l_k} are nested, then, from Equation (4.20), we can check that the level set $\mathbf{u}^* \in \mathcal{C}^S$ defined for all $p \in S$ by

$$u_p^* = \max\{i \in \mathcal{M} \mid u_p^{*l_i} = 1\} \quad (4.23)$$

minimizes Equation (4.20).

According to [Darbon and Sigelle \[2006\]](#), if the condition Equation (4.22) holds for unary terms, a minimizer of E can be deduced from all the minimizers of $\{E^{l_i}\}_{i \in \mathcal{M} \setminus \{0\}}$. In particular, the condition Equation (4.22) is true for convex unary terms and when $E_{p,q}$ is the total variation (or here the perimeter of u^{l_i}), which is the case in our applications for the data term.

Due to the monotone condition Equation (4.22), binary solutions are nested. The divide-and-conquer process proposed in [21] takes advantage of this property and via the tree traversal of a tree of split values, defining a set of independent binary problems. It allows us to decrease drastically the number of s-t minimum cuts. The key point for reducing the number of used graph cuts is the definition of a hierarchical tree having K leaves and whose nodes encodes the binary subproblems.

Each subproblem of Equation (4.21) is solved by max-flow/min-cut algorithms. If (4.23) holds, instead of computing cuts for every pair of labels $\forall (\alpha, \beta) \in \mathcal{C}^2$, or for every pair of consecutive labels in direct order, we can maximize the information received from each cut by splitting by two the available label set for each site. This is the divide and conquer process proposed in [Ribal et al. \[2018\]](#): Similarly to a coarse to fine resolution, we build a tree of split values that defines a set of independent binary problems.

Because the level sets are nested, the binary solution minimizing the energy of this first subproblem acts like a thresholding of the optimal solution by indicating to which subset of label belongs the optimal label of each site. In particular, it means that testing all the threshold labels for all the sites is not necessary. Instead, the sites that were labeled 1 in the first subproblem are tested for a higher threshold value, and sites labeled 0 are tested for lower values of the threshold label. This is made possible by a modification of the graph: After each subproblem, the edges that link two sites labeled differently are removed, and a data cost penalty is added to one of the terminal links weight of the sites, depending of their labeling.

The initial graph of each step $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i, c)$ is split into two graphs according to the labeling \mathbf{v} : $\mathcal{G}_\alpha = (\mathcal{V}_\alpha, \mathcal{E}_\alpha, c)$ and $\mathcal{G}_\beta = (\mathcal{V}_\beta, \mathcal{E}_\beta, c)$.

One of the subsets $\mathcal{V}_\alpha = \{p \in \mathcal{V}_i \mid v_p = 1, \text{ or } p \in \{s, t\}\}$ only contains sites whose optimal labels are higher or equal to the threshold α , and the second one $\mathcal{V}_\beta = \{p \in \mathcal{V}_i \mid v_p = 0, \text{ or } p \in \{s, t\}\}$ only contains sites whose optimal labels are lower than the threshold α (lower or equal to $\beta = \alpha - 1$). Edges of those graphs are the edges of the initial graph with the set of edges linking both graphs removed, the latter one being also the edges of the min-cut in \mathcal{G}_i : $\mathcal{E}_\alpha = \{(p, q) \in \mathcal{E}_i \mid p, q \in \mathcal{V}_\alpha\}$, and $\mathcal{E}_\beta = \{(p, q) \in \mathcal{E}_i \mid p, q \in \mathcal{V}_\beta\}$.

We note that those subgraphs are disconnected and can be treated separately. In the general case, every label α may be tested, and therefore $\#\mathcal{C}$ graphs may be built, but in practice, those subgraphs are smaller and their processing per step is completely parallel. Said differently, all the subproblems of a given step of the algorithm can be computed simultaneously by a single minimum cut in a graph of the size of the initial one \mathcal{G} .

With these properties and for comparison with heuristics of α -expansion and α - β swap, the number of minimum cut required for dyadic search actually reduces to $\log_2(\#\mathcal{C})$ and ensures convergence, without iterations.

4.2 Functional minimization

4.2.1 Dictionary

With dictionary neighborhood, the functional is modified in Equation (3.7) as follows:

$$F_2(\mathbf{u}, V) = E_1(\mathbf{u}) + \alpha E_2(\mathbf{u}, V) + \beta E_3(V).$$

The global energy F_2 depends on two terms at the same time, the segmentation \mathbf{u} and the neighborhood V , but it is possible to split the minimization problem into two subproblems by adopting an alternative minimization: Firstly the minimization of the energy F_2 with fixed segmentation \mathbf{u} and variable neighborhood V , and secondly the minimization of F_2 with fixed neighborhood V and variable segmentation \mathbf{u} . At each step, the energy is minimized with respect to the free variable, meaning that F_2 is guaranteed to decrease. With such a formulation however, since the energy is not convex, the convergence to a global minimum is not ensured. However, because both the set of possible neighborhoods and the set of possible segmentations are finite, the energies E_1 , E_2 and E_3 are discrete. With this scheme, F_2 is a discrete and decreasing function, and therefore the convergence is ensured. In particular, this is true because our minimization is deterministic, and favors the initial configuration of the variable to every other equal-energy configurations, so the output cannot oscillate between multiple local minima of equal energies.

Thanks to these properties, we are guaranteed that after an initial segmentation has been fixed, estimating the neighborhood and refining the segmentation iteratively will not lead to a higher energy situation than with a single straightforward estimation, and may even prevent oscillatory behaviors.

If we truly consider the problem as two different functional, we can recall that it is a double regularization: For segmentation, E_1 and E_2 play respectively the role of data energy term and smoothness energy term for \mathbf{u} , but for neighborhood, it is E_2 and E_3 that endorse those roles. The chance is that the smoothness term for \mathbf{u} , is also the data energy term for V allows such formulation.

This iterative procedure is summarized in Algorithm 4.

- Initialize \mathbf{u}^0 , $n = 0$.
- Repeat while $\sum_{s \in \mathcal{S}} \mathbb{1}_{\{\mathbf{u}_s^n \neq \mathbf{u}_s^{n-1}\}} > \varepsilon |\mathcal{S}|$
 - Minimize Equation (3.7) with \mathbf{u}^n fixed to compute with graph cut

$$V^n \in \underset{V}{\operatorname{argmin}} \alpha E_2(\mathbf{u}^n, V) + \beta E_3(V),$$
 - Minimize Equation (3.7) with V^n fixed to compute with graph cut

$$\mathbf{u}^{n+1} \in \underset{\mathbf{u} \in \mathcal{C}^{\mathcal{S}}}{\operatorname{argmin}} E_1(\mathbf{u}) + \alpha E_2(\mathbf{u}, V^n).$$
 - Increment n

Algorithm 4: Algorithm used for approximating a minimizer of F_2 , in the configuration where the best neighborhood is computed as the configuration of elements from a dictionary that minimizes the modified functional. $\varepsilon \in \mathbb{R}_{>0}$ is a small real number.

The optimization algorithm we use for minimizing the energy for both steps of the algorithms is graph cuts. As explained in Section 4.1.1, graph cuts provide an efficient way for solving minimization problems in the way they are formulated in Equation (3.7).

4.2.2 TV and RORPO based neighborhood

The same idea applies with TV and RORPO but there is a crucial difference: The functional (Equation (2.3)) is not modified for incorporating the vesselness operator, because it has no formulation as an energy data term that would match the term E_2 . Instead, we trust the estimation of those robust operators from a set of data including an initialized segmentation \mathbf{u} . What remains consistent with previous description is that we decide to treat alternatively the two parts of our problem, that is to say that on the one hand we estimate the neighborhood from fixed segmentation, and on the other hand we regularize the segmentation with anisotropic neighborhoods fixed in advance.

- Initialize \mathbf{u}^0 , $n = 0$.
- Repeat while $\sum_{s \in \mathcal{S}} \mathbb{1}_{\{\mathbf{u}_s^n \neq \mathbf{u}_s^{n-1}\}} > \varepsilon |\mathcal{S}|$ or if $n < N_{iMax}$:
 - Estimate \mathbf{g}^n from \mathbf{u}^n and the available data,
 - Construct V^n from \mathbf{g}^n ,
 - Minimize Equation (2.3) with V^n fixed to compute with graph cut

$$\mathbf{u}^{n+1} \in \underset{\mathbf{u} \in \mathcal{C}^{\mathcal{S}}}{\operatorname{argmin}} E_1(\mathbf{u}) + \alpha E_2(\mathbf{u}, V^n).$$
 - Increment n

Algorithm 5: Algorithm used for approximating a minimizer of F , in the configuration where the best neighborhood is computed from a vesselness operator or any other guidance map. $\varepsilon \in \mathbb{R}_{>0}$ is a small real number, and N_{iMax} is a limit to the number of iterations.

Indeed, with such a formulation we loose the guarantee of convergence established in previous section: It is always possible that the $(\mathbf{u}^n)_{n \in \mathbb{N}}$ oscillates endlessly due to complementary oscillations of $(V^n)_{n \in \mathbb{N}}$. For this reason, the formal definition of Algorithm 5 includes an additional stopping criterion when the number of iterations exceeds an arbitrary fixed value $N_{iMax} \in \mathbb{N}$.

Additional options for improving the convergence of the minimization algorithm may be considered. For instance, the module used for orientation estimation may be modified with the value of the iteration step n of the algorithm. In the module that computes the orientation and saliency of thin structures, scale parameters such as σ_T in TV and L in RORPO could be tuned in order to progressively refine the segmentation, thus defining a coarse-to-fine approach. In this way, the largest structures would be detected in the first steps of the algorithm, while the following steps would allow to refine the segmentation in areas with higher tortuousness. However, evaluating the convergence and efficiency of such approach in an iterative multi-scale context has been left for future works.

4.2.3 Direct neighborhood estimation

In practice, and because of this specific complexity of the stopping criterion, our contribution focuses more on the effect of an unique iteration with $N_{iMax} = 1$. We indeed carried minimal experiments on the iterative process, and leave the field open for future works. We are therefore able to propose a generic formulation, but the main contribution and the majority of our researches deal with a non-iterative minimization of \mathbf{u} . For doing so, we initialize \mathbf{u}^0 , follow the same steps of Algorithm 5 but stop when $n = 1$. Instead of refining the neighborhood, we only estimate it once. The estimated neighborhood is imperfect, subsequently to noise on the initial blind segmentation \mathbf{u}^0 , but we rely on the robustness of the vesselness operator with respect to noise and segmentation errors so that V is accurate enough for improving the resulting segmentation.

4.3 Implementation

4.3.1 Superpixel generation

We have tested a small set of superpixels that shared desirable properties, were publicly available, and matched our theoretical background. We have selected them among comparative studies [Stutz et al. \[2018\]](#); [Wang et al. \[2017\]](#), and implemented them in our program. The five candidates we focused on are [Felzenszwalb and Huttenlocher \(FH\)](#), [Simple Linear Iterative Clustering \(SLIC\)](#), [Superpixels Extracted via Energy-Driven Sampling \(SEEDS\)](#), [ETPS](#) and [Waterpixels \(WP\)](#).

Images in Figure 4.9 show typical examples of superpixels features depending on the used superpixel algorithm that ease the choice of this latter. We select two different values for the number of superpixels $\#S$, 2500 and 10000 for the whole image, but we do not lead an intensive investigation of the optional other parameters of the algorithms that we let to their default values. We also note that FH do not propose a parameter for setting the number of superpixels.

However, we also question the benefits of some pre-processing aimed at improving the superpixels we obtain: We implemented [Zou et al. \[2012\]](#)'s shadow removal algorithm, as well as median filtering and Gaussian filtering, on the second row of each stack of superpixel images in Figure 4.9. While the shadow removal improves the results of the segmentation, the shape of the superpixels is mainly influenced by the Gaussian and median filtering, since they tend to reduce the contrast of the textures of the image.

We can see that the implementation of FH [Felzenszwalb and Huttenlocher \[2004\]](#) from [Stutz et al. \[2018\]](#) (first column) is very poor in terms of consistency of superpixel size, i.e. the variation of area of the superpixels is high, particularly in the more homogeneous regions. This is caused by the way FH superpixel are computed. First, the elements processed by the algorithm is initialized from the set of pixels. Then, these elements are progressively merged as long as a measure of the minimum internal difference in a pair of adjacent elements is higher than a measure of the difference between these two elements. While the measure of the minimum internal difference inside a pair of elements includes a term with a parameter that may favor larger superpixels, there is no warranty on the sizes of the superpixels produced. This explains the high variability in size depending on the content of the image that makes the implementation of anisotropic neighborhoods at superpixel level more complicated.

In the case of the ground truth, the whole crack is composed of only one superpixel, and the same behavior is observable in small parts of the crack in the original image, whatever the number of superpixels $\#S$. We note that the superpixels of the ground truth based segmentation have already been made considerably more regular thanks to the addition of a slight noise on the input image. These observations let us think that FH is a probably bad choice for testing our segmentation algorithm, especially on simulated images. This have been confirmed in early experiments, whose data have unfortunately not been stored.

The obtained results using SLIC [Achanta et al. \[2012\]](#) (second column) exhibits one of the major drawbacks of the naive clustering-based approach: Disjoint pixels require a postprocessing step to enforce connectivity of superpixels. That being said, the compactness parameter of the SLIC algorithm could reduce this artifact, that is particularly visible in original image and progressively disappears when the presence of texture decreases. For these reasons, we discard this algorithm for the moment.

The next two columns SEEDS [Van den Bergh et al. \[2012\]](#) and ETPS [Yao et al. \[2015\]](#) present superpixels that are based on the formulation of an energy term. The image is initialized as a regular grid partition, and pixels are exchanged between regions to iteratively minimize this energy. Both seem to offer more regular superpixels, in terms of shapes and layout. They are still very sensitive to the ground's texture, which creates non-smooth superpixel borders when using the original image as an input, but become smoother with the preprocessed image [Aldea and Le Hégarat-Masclé \[2015\]](#).

In Figure 4.10, [Yao et al. \[2015\]](#) present an illustration of ETPS superpixel construction. The superpixels are obtained from iteratively minimizing an energy on a grid that is progressively sub-

divided until reaching pixel level. This is a coarse to fine approach, in which at each step, the superpixels may evolve such that the energy, that involves terms encouraging spatial consistency and compacity, radiometric homogeneity, and forbid high variations in superpixels sizes, is minimized.

We expect a suitable superpixel algorithm to be able to define superpixels inside the objects we want to detect. In other words, superpixels must be small enough to fit inside the objects of interest, or adaptive enough to shrink into those objects. Combined with the preference we have for regular superpixels and the interest of lowering the number of segments, we can indicate that $\#S = 2500$ is a good lower bound for the number of superpixels, since increasing this value does not seem to improve drastically the performances with respect to the aforementioned criteria.

In terms of speed, ETPS, SEEDS and SLIC run twice faster (around 6.5 seconds) compared to FH (around 12 seconds) when computing $\#S = 2500$ superpixels on an image of 800 per 600 pixels.

Since ETPS seems to give exploitable results with respect to our constraints, we opted for this algorithm to conduce our first experiments.

Later, in the application to SFF, we have introduced WP [Machairas et al., 2015], for providing an illustration of the impact of superpixel generation on the quality of the anisotropic regularization. WP generate homogeneous superpixels from a set of seeds using a distance function based on a Lab-gradient, in the CIELAB colorspace. Each seed is associated with a unique connected component by applying the watershed transformation. Those superpixels exhibit a good adherence to objects boundary, by construction, but also have a regular positioning in space since the initial seeds are selected on a regular grid. Figure 4.11 shows an example of FH, WP and ETPS generated on one image from our dataset in SFF. We note that FH still produces highly variable sizes and shapes of superpixels, which combined with the need for regular superpixels for our anisotropic neighborhood confirms our preference for other algorithms like WP and ETPS.

4.3.2 Programming

Neighborhood precomputation

Since the regular lattice is lost with superpixels, it is important to stress that when we refer to the sites by their indexes, the direct relation between the index and the location of the site is not known anymore. To be more precise, when seeking for a neighbor, one has multiple choice: Either, try all the available sites, and test each one of them, or either do this research at pixel level in a limited area, by testing if the pixel belongs to a site, and then testing this site. However, both are time consuming, since the extensive test of $\#S$ should be avoided, and the test at pixel level, even in a restricted area, is longer than the same test at site level.

For this reason in practice, we precompute a larger disc based isotropic neighborhood that restricts the search of potential neighboring superpixels, before computing our anisotropic neighborhoods. In some circumstances, we may miss some superpixels that would otherwise be part of the neighborhood, but it allows for a clear reduction of the computing time with around $\mathcal{O}(k\#S)$ operations instead of $\mathcal{O}(\#S^2)$, where $k \in \mathbb{N}$ is the average number of sites in the restriction map, per site.

Graphical User Interface

Image processing is an area of science in which we manipulate fields that contain a large number of variables. To operate these variables, we often need to simultaneously display multiple fields for making the phenomenon understandable at one sight. For these reasons, we developed our own tool for monitoring each step of the process and easily visualize the processed data. While being remaining relatively simple, this approach required a bit of work in C++ language using Qt as a graphic interface. Due to early choices in the thesis implementation, for instance the will to keep Qt an optional dependency, the interface is mostly focused on visualization. Therefore, interactive computation remained limited, but the interface still allowed us to explore lots of situations during the experiments.

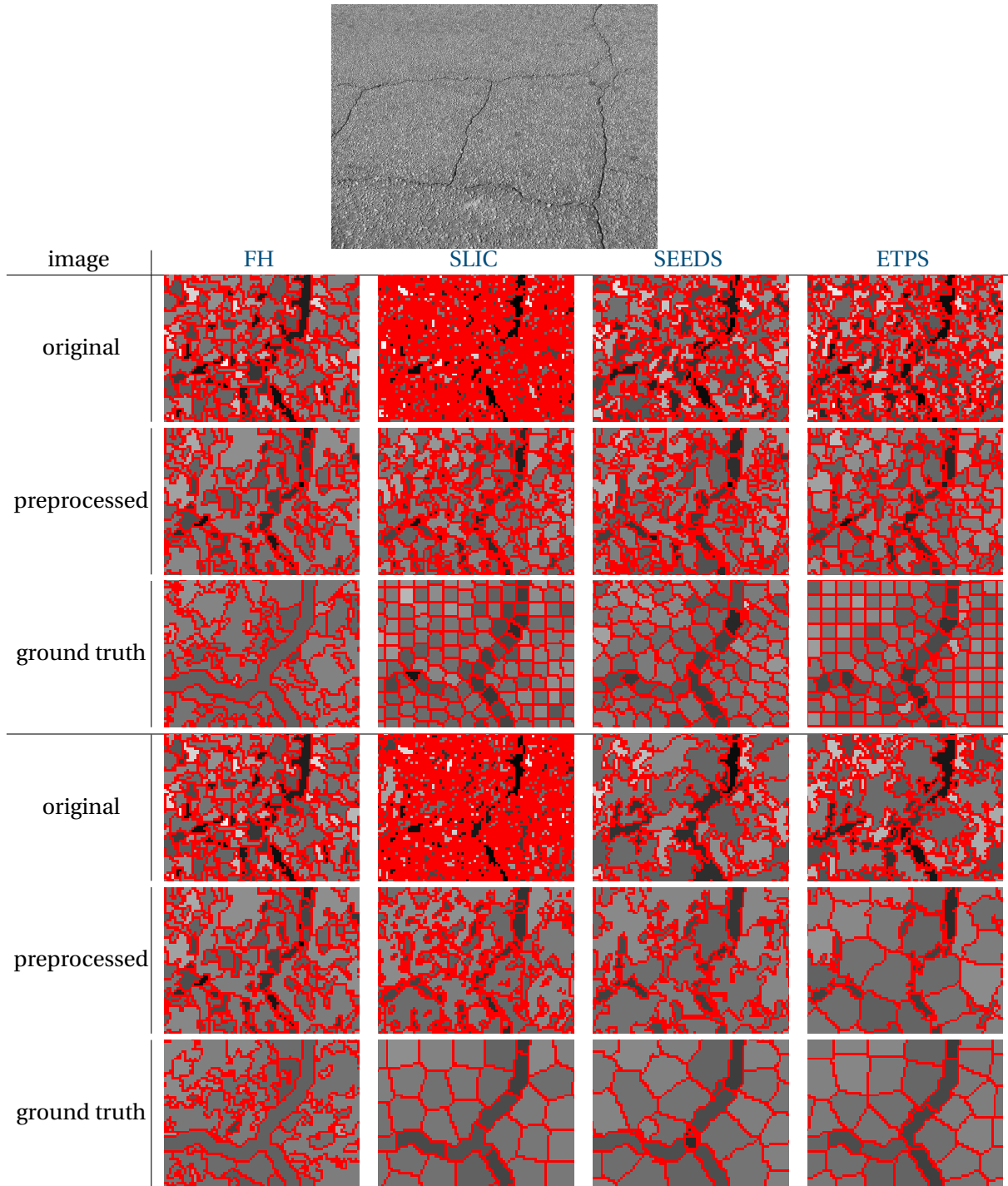


Figure 4.9 – Superpixels obtained on the crack image “6193” of our dataset (above the Table). The input images, zoomed on the lower right crack junction, are shown in Figure 2.6. Here, the number of superpixels generated are 10000 for the upper half and 2500 for the lower half. We present four algorithms: FH proposed by Felzenszwalb and Huttenlocher [2004], SLIC by Achanta et al. [2012], SEEDS by Van den Bergh et al. [2012] and ETPS by Yao et al. [2015]. Since crack images are very textured, we also propose a pre-processing of the image based on Gaussian and median filtering, for smoothing the results, and also show as a comparison the behavior of the superpixel computed from the ground truth image. After a few experimentation, we have decided to stick with ETPS superpixels for our analysis.

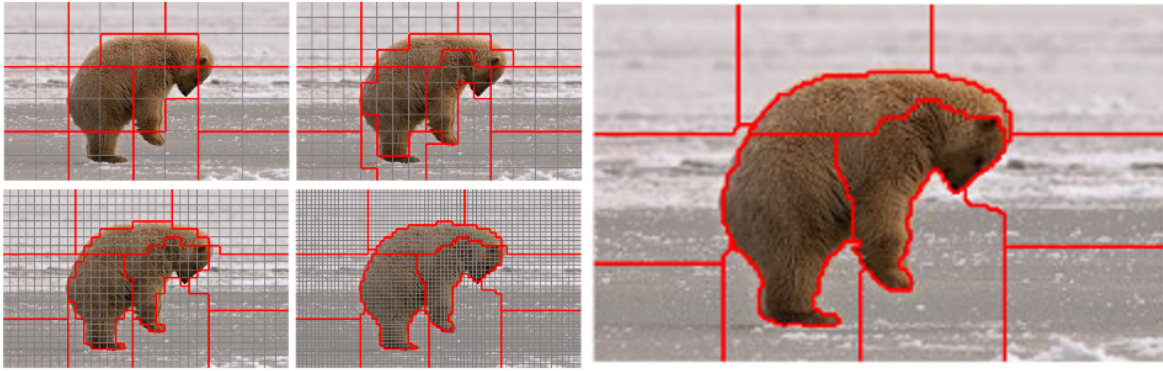


Figure 4.10 – ETPS superpixel construction (courtesy of Yao et al. [2015]). ETPS minimizes an energy by moving the set of boundaries of an initial set of superpixels on a grid with a finer and finer resolution. In this example, the grid initially has 56 regions (top left image) and each region is subdivided until the obtaining of the final segmentation (image on the right). The energy terms include geometric and radiometric homogeneity constraints, as well as size, perimeter and compacity constraints.

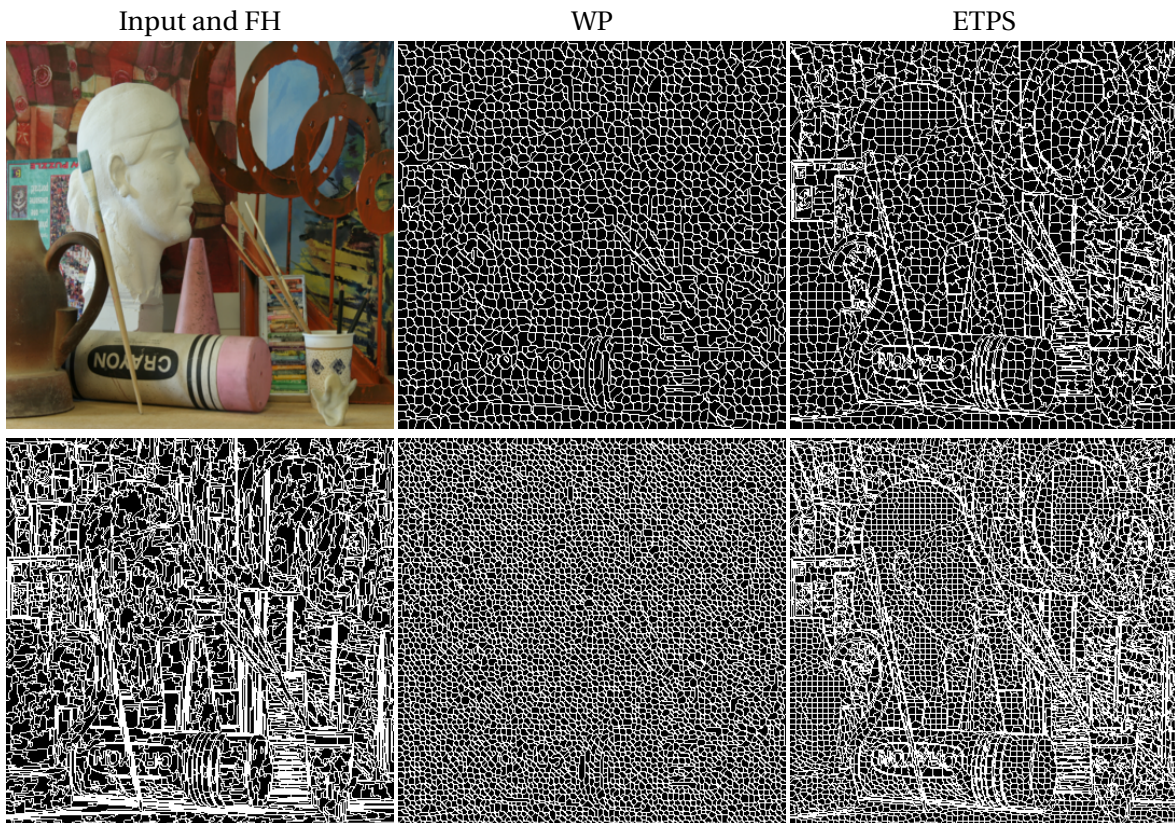


Figure 4.11 – Example of superpixels computed on Art image (input image in the first row, left column), with FH [Felzenszwalb and Huttenlocher, 2004] superpixels (left column, second row), WP [Machairas et al., 2015] (second column) and ETPS superpixels [Yao et al., 2015] (last column), both for 2000 and 5000 superpixel images, respectively in the first and second row. The high variability of FH sizes and shapes is self explanatory when looking at the bottom left image. In comparison, the positioning of WP and ETPS superpixels is more regular, due to the seed positioning for the former, and to the grid-like approach for the latter. Visually, the good properties of boundary adherence of ETPS superpixels encouraged us to use this algorithm first.

For instance, for SFF application, a 3D viewer has been developed for analyzing visually the volume of data, see Figure 4.12. Tensor fields from TV, and more precisely the evolution of the fields of eigenvectors and eigenvalues, are examples of such observations. This tool also allowed us to perform some parameters tuning manually in neighborhood construction, by showing interactively the behavior of neighbors of a site and computing at the same time some statistics about global neighborhood performances (see Section 6.3.2 for more details on neighborhood quality estimation).

Git repository

The code used for RORPO and TV computation in this PhD will be available on github² as a contribution. The GUI and other tools may be published as well on this repository, after some reviews and commenting on the code.

4.3.3 Parameters setting

The introduced model depends on several parameters. Some of them are free, and will influence the performances of our approach and require to be tuned to the application, while other are fixed by our model. In this section we discuss roughly the way we tuned these values according to our model and experiments, and list them in Table 4.2. Some parameters however depend more specifically on the application, and we mention them in their relative chapters.

First, we implemented multiple algorithms of superpixels and decided visually of their quality with respect to our application. These comparisons are described in Section 4.3.1. Our experiments lead us to favor ETPS superpixels, that we implemented with a few pre-processing in our binary examples with cracks and simulated images, whose textures are very contrasted. The constraints with those pre-processing was to preserve the thin cracks, and explain the small windows sizes we used. With SFF application, we did not filter the images the same way, however, we still increased the value of the smooth weight parameter of the algorithm to a hundred, that gave visually better and logically smoother results.

Energy-based vesselness only integrates one coefficient of regularization, β , for which we empirically fixed three values 0.5, 1 and 2, for seeing its effect on the guidance map regularization.

The other vesselness operators include much more parameters, which makes any extensive study more complicated. However, since they gave better results in our experiments, we spent some time tweaking the values of the parameters and decided of what is listed on the Table: after computing a wide range of values of σ_T in TV for instance, we fixed $\sigma_T = 30$ for comparison with the RORPO in the SFF application. The kernel curvature coefficient has been fixed manually so that in the kernel decay function expression that integrates a negative exponential with two terms, the order of magnitude of the first term $\nu\phi^2$ would be comparable with the squared arc of circle length. Then in the case of SFF, since the resolution in depth controlled by the number of defocused images is not the same as the resolution in the image plane controlled by the image sensors, we added a coefficient in our preliminary trials so that ball kernels would seem reasonably spherical. The latter kernel is computed as an approximation of an integral, which was inspired by the Matlab code existing for TV³, and we fixed the number of steps computed per axis to 16.

With the RORPO, we have fixed six orientations for space sampling, since it can already produce a lot of orientations with the guidance map orientation refinement. The angle threshold have been fixed a bit wider than with Merveille et al. [2018], to reduce the impact of the irregular lattice on the path opening with larger overlapping orientations. Therefore, the angle threshold is set to four thirds of the angle between consecutive opening orientations. The minimal length for path opening has been set to 3 times the radius of an average site in the image, which allows relatively

²Code available in github repository <https://github.com/cstoribal/SANTOS> (Superpixel Anisotropic Neighborhood for Thin Objects Segmentation).

³Code currently available publicly on <https://fr.mathworks.com/matlabcentral/fileexchange/21051-tensor-voting-framework>

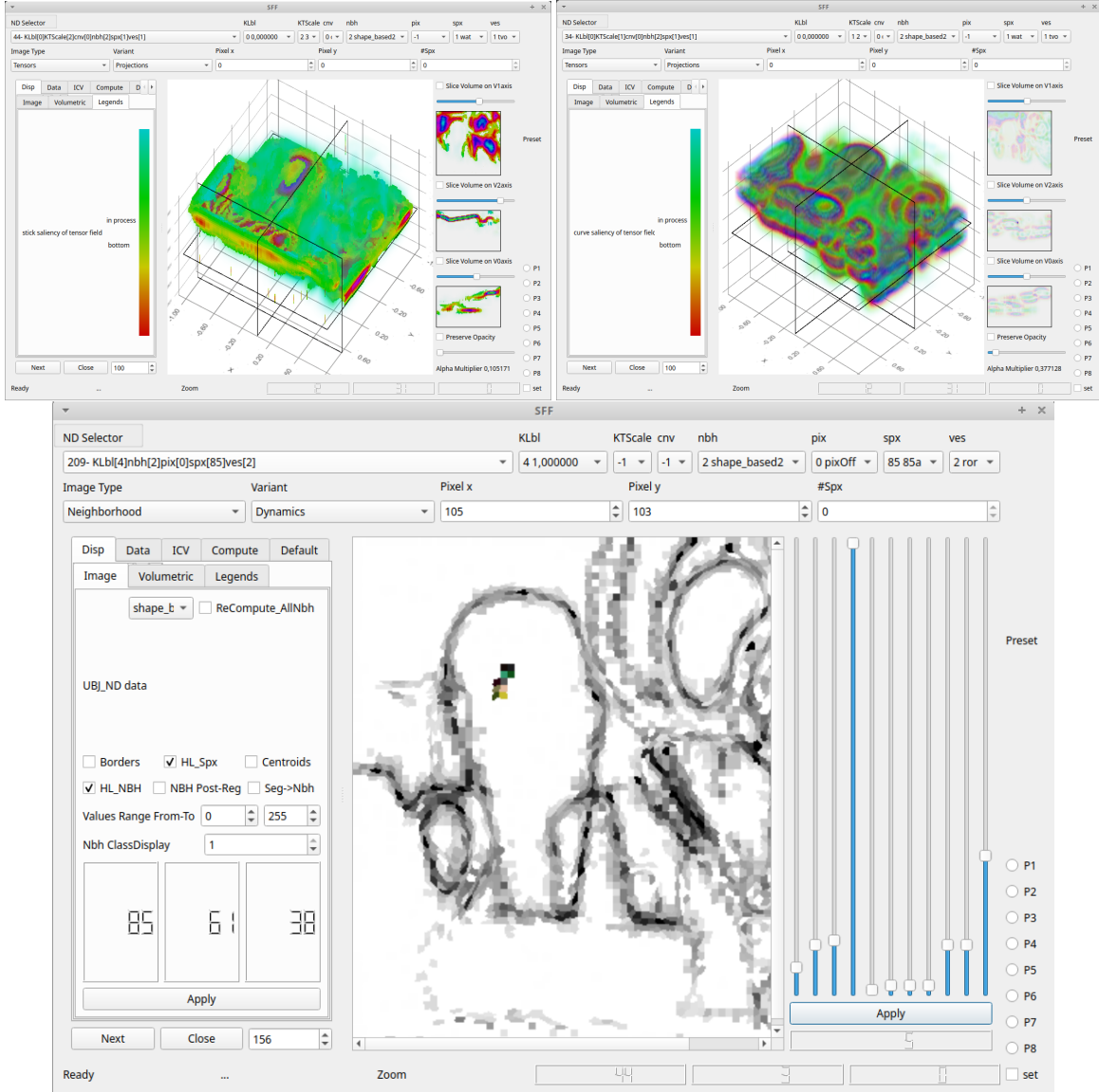


Figure 4.12 – Graphical User Interface: Examples of tools used for easing the understanding the implementation of anisotropic neighborhoods. On the first row, the 3-dimensional plot allow to display in false colors the values of the stick (left image) or plate (right image) saliencies projected on the axis e_2 . Additionally, this plot allow to visualize 2D slices of the data at any coordinates. However, we did not used nor presented any maximum saliency extraction method for the derivation of guidance map from TV, since it has not produced any convincing results yet. In the second row, this is an example of an interactive tool to construct neighborhoods and explore manually the effect of parameters tuning. It offers visual results for qualitative evaluation and computes some measures for quantitative evaluation as well, and has been of great use for understanding some aspects of our approach.

| Context | Parameter | symbol | values (if fixed) | |
|--|--|-------------------|------------------------------|-----------------|
| | | | Binary | SFF |
| Supapixel generation | Type of superpixel | - | ETPS | |
| | Number of sites | $\#\mathcal{S}$ | $\in [2500, 5000]$ | |
| | ETPS smooth weight | - | 100 | |
| | Median filtering window size | - | 3×3 | - |
| | Morphological opening kernel size | - | 3×3 | - |
| | Morphological closing kernel size | - | 3×3 | - |
| Energy based vesselness regularization | | β | $\in [0, 2]$ | - |
| Vesselness TV | Scale initial tensors from data | - | no | |
| | Initialize from sites' shapes | - | no | |
| | Kernel scale | σ_T | $\in [25, 200]$ | 30 |
| | Kernel curvature | ν | 2×10^5 | |
| | Kernel depth scale | - | N.A. | 5 |
| | Kernel discretization steps | $I = J$ | 16 | |
| Vesselness RORPO | Kernel maximum ϕ | - | 45° | |
| | Number of orientations | - | 6 | |
| | Opening angle threshold | ϕ_T | $4\pi/18$ | |
| | Path opening length | L | $3\sqrt{\hat{A}_S}$ | |
| | Anisotropic orientation threshold | $i_{\mathcal{R}}$ | 2 | |
| | Normalization of input data | - | - | no |
| Neighborhood construction | Pixel adjacency | - | 4-adjacency | |
| | Weights normalization | - | yes | |
| | Disc radius | R | $\sqrt{20\hat{A}_S/\pi}$ | |
| | Guidance map isotropy threshold | Γ | 0.05 | |
| | Shape type | - | ellipsis | |
| | Shape area | A_V | $7 \times \hat{A}_S$ | |
| | Shape ellipsis eccentricity | ϵ | 0.9 | |
| | Dictionary shape set size | ι | $6 + 1$ | |
| | Isotropic shape relative scale | - | 2 | 1 |
| | Target based cost function weighting | η | $5 \times 10^{-3}\sqrt{A_V}$ | 30 |
| | Cardinal based cost function weighting | η' | 5×10^{-3} | 100 |
| | Cardinal based path length | K | 4 | 3 |
| | Regularization coefficient | α | $\in [1/64, 1]$ | $\in [1/8, 64]$ |
| | Stopping criterion | ϵ | - | |
| | Maximum iteration | N_{iMax} | 1 | |

Table 4.2 – Table of parameters, for listing the main parameters that we have tuned during our experiments. Most of them have been slightly tuned by trial and error, sometimes using our graphical user interface (GUI) for finding better combinations of parameters. In practice, we have implemented more internal parameters, such as boolean switches for trying alternative computations, but we have limited their appearance for more clarity.

short paths to be conserved in the opening, but gives good results. For encouraging the detection of only thin structures however, only sites being member to up to two different paths are considered belonging to anisotropic structures: The orientation threshold $i_{\mathcal{R}}$ has been set to 2. In the case of 3D spaces with RORPO computed on 2D images, we tried the normalization of the input data with respect to each site of the image, but it did not produce any improvement.

Finally, there are a few parameters also that impact the neighborhood construction. First, to derive adjacency at site level, we base on 4-adjacency at pixel level, since the size of these sites and the length of their borders makes the difference with 8-adjacency or more complex adjacency relationship unnoticeable. For providing the weighting function W , we normalize the weights of the neighbors of a site so that they sum to one: This is a design question that gives a better understanding of the regularization coefficient α , that can be considered as a parameter.

Then, we set the parameters of each neighborhood specific implementation as depicted in the table. More generally, with shape (and disc) neighborhoods, the issue is to ensure that the center of mass of nearby sites may belong to a neighborhood that would seem regular if possible, and to fit in the thin structures at the same time. Typically, larger neighborhoods produce more regular topology, since a large radius R or a shape area A_V , will help to avoid missing a close neighbor. Conversely, thin neighborhoods with small areas and high excentricity will tend to “miss” the location of the barycenter of close sites that would however be good potential neighbors, and worse, some sites may also have no neighbors for the same reasons. In isotropic cases with crack detection, we have empirically seen that increasing the size of the isotropic shape helped with the regularization of homogeneous areas. We also tried implementing rectangular and conic neighbors as a replacement of ellipsis based neighborhoods, but it did not seem to improve the results.

Similarly, we have realized experiments with path based neighborhoods, and tuned the parameters by hand while computing interactively the precision of these neighborhoods. The conclusions of these empirical observations were that, in SFF, allowing for a good flexibility of the neighborhood with respect to the expected orientation and reducing the length of cardinal based neighborhoods to $K = 3$ (corresponding to 4 neighbors) seemed to produce the best results.

We note that the values used for weighting parameters η and η' are very different depending on the application. This is because in the experiments we realize with thin structure detection, I is expressed in range $[0, 1]$, while in SFF, the input image I in Equation (3.17) is the sharpness profile, constituted of sum of Laplacians of images that we have not normalized. However, in both situations, we note that η and η' values correspond to high sensitivity to the image content component, and a rather reduced sensitivity to orientation.

We compute our experiments for multiple values of α , that have been fixed so that both under-regularized and over-regularized situations are encountered, and finally, we limited ourselves to one iteration of minimization of the functional, therefore we did not fix any stopping criterion in our experiments.

In the following chapters, we now specify the implementation for both applications and present a few results in the meantime.

Chapter 5

Thin structure detection

Contents

| | |
|---|-----------|
| 5.1 Context | 70 |
| 5.2 Binary implementation | 70 |
| 5.2.1 Energy terms | 70 |
| 5.2.2 Parameter tuning | 71 |
| 5.3 Numerical experiments | 72 |
| 5.3.1 Guidance map estimation | 72 |
| 5.3.2 Qualitative and quantitative evaluation | 72 |
| 5.4 Conclusions | 76 |

5.1 Context

In this chapter, we compare results obtained using our neighborhoods with results using isotropic neighborhoods, in two examples: detection of geometrical drawn lines in a simulated image, and crack detection on some actual gray-level images. Both detections result in a binary segmentation where thin structures should be highlighted. For segmentation of crack images, [Nguyen et al. \[2011\]](#) motivates the need for a detection method that takes advantage on free-form anisotropic path-based neighborhoods. We note that those neighborhoods share some similarities with the neighborhoods introduced in Section 3.3.3. However, the approach remains specific to cracks and textured backgrounds and we therefore decided to implement our more generic approach as a challenge and as a proof of concept.

The simulated image is a grayscale image, with arbitrary shapes and textured noise on it. This image was generated from a grayscale image with structured drawings, such as lines, arcs of circles that provides a ground truth from thresholding. Then, we simulate noise by manually adding artifacts and shadows to the image.

Crack detection consists of the binary segmentation of real images from the CrackTree dataset ([Zou et al. \[2012\]](#)), which includes the ground truth images. In these images, the cracks are thin structures over a highly textured and noisy background, for instance some asphalt road or a concrete wall. Even with the preprocessing for removing shadows proposed by [Zou et al. \[2012\]](#) and some background subtraction methods, the texture of such surfaces makes any segmentation by thresholding unsuitable, which motivates the need for regularization. Note also that the probability density of the two classes “crack” and “background” are different and that in terms of area cracks only represent a very small amount of sites. Therefore, they tend to disappear with regularization. For addressing this issue, we encourage over-detection of cracks at data level (without regularization). Figure 2.6 and Figure 5.2 show an example of each scene type and the associated ground truth. We note that another crack dataset [[Majidifard et al., 2020](#)] exists and presents a more complete set of images with more labels, such as longitudinal, reflective or alligator cracks. However, since this dataset has been proposed during the course of this PhD and after our early experiments on the subject, we continued our experiments with the CrackTree dataset that was already used in previous experiments in our laboratory.

In the next sections, we specify the energy terms we use for this application (Section 5.2.1), as well as some specific parameters of implementation (Section 5.2.2). The evaluation itself (Section 5.3) consists of the visual comparison of the guidance map obtained with energy-based and TV models, and then a numerical comparison of performance obtained with the best pair of parameters (α, σ) for TV.

Finally, note that, in the framework of this PhD, this first application of our contributions for anisotropic neighborhood was only considered as a toy-application since, unfortunately, we had not enough time for further investigation.

5.2 Binary implementation

5.2.1 Energy terms

The data fidelity term $E_1(\mathbf{u})$ in the functional $F(\mathbf{u}, V)$ (see Equation (2.3)) is the energy term derived from the likelihood $\mathbb{P}(I|\mathbf{u})$. In the application to crack detection, the model for assigning a site to either the background or the crack is often based on the assumption that a crack is shadowed by the material in the surroundings and therefore has a lower luminosity than the background. For usual gray-level images (as CrackTree dataset ones and simulated one), image intensities are usually assumed Gaussian-distributed for each class $c \in \mathcal{C}$ with mean value $\mu_c \in \mathbb{F}$ and standard deviation $\sigma_c \in \mathbb{R}_{>0}$. The data term E_1^s for any superpixel $s \in \mathcal{S}$ and any label $u_s \in \mathcal{C}$ can be written:

$$E_1^s(u_s) = \frac{\|I(s) - \mu_{u_s}\|_2^2}{2\sigma_{u_s}^2} + \log(\sigma_{u_s}),$$

and, assuming conditional independence (cf. Section 2.3.2) for any $\mathbf{u} \in \mathcal{C}^{\mathcal{S}}$, E_1 in Equation (2.3) is:

$$E_1(\mathbf{u}) = \sum_{s \in \mathcal{S}} E_1^s(u_s). \quad (5.1)$$

The way class parameters have been set is specified in the next section about parameter tuning.

For the second energy term, $E_2(\mathbf{u}, V)$ with $\mathbf{u} \in \mathcal{C}^{\mathcal{S}}$ and $V \in \mathbb{V}$, we opt for the Potts model Wu [1982], weighted according to the strength of interaction between neighboring sites:

$$E_2(\mathbf{u}, V) = \sum_{s \in \mathcal{S}} \sum_{t \in V(s)} W(s, t) \mathbb{1}_{\{u_s \neq u_t\}},$$

where $\mathbb{1}_{\{a \neq b\}} = \begin{cases} 1 & \text{when } a \neq b, \\ 0 & \text{otherwise.} \end{cases}$ and $W : \mathcal{N} \rightarrow \mathbb{R}_{>0}$ is the weighting function specified right after.

In the case of the neighborhood of Stawiaski and Decenci re [2011], i.e. the weighting function W is defined for any pair $(s, t) \in \mathcal{N}$ as $W(s, t) = \frac{\partial(s, t)}{\partial(s)} \in]0, 1]$, where $\partial(s, t)$ and $\partial(s)$ denote the common boundary between s and t and the perimeter of s , respectively. Now, such a definition requires that s and t have a common boundary, i.e. are adjacent.

Then, alternatively, we propose a weighting function that overcome such a constraint since several of our neighborhood constructions may involve non adjacent pairs of sites within the cliques. Specifically, we propose to define for any pair $(s, t) \in \mathcal{N}$ the weighting function W as $W(s, t) = (\#V(s))^{-1} \in]0, 1]$.

The Potts model maintains the property of submodularity and the data fidelity term E_1 is convex. Since in our case $\#C = 2$, then according to Kolmogorov and Zabih [2004], we know for sure that the energy function defined in Equation (2.3) can be exactly minimized.

5.2.2 Parameter tuning

In our implementation, the image intensities are normalized in $[0, 1]$.

The parameters of our two classes, background and crack, have been firstly set after measuring the mean value and standard deviation of their grey level from the ground truth first. However, first experiments showed us that non stationarity of the road texture induces very numerous FP detections in the bottom of the image while on the top of the image the FN cases become a majority. Actually, the angle formed by the optical axis with the normal of the surface imaged induces a variation in both the lighting and the size of the texture relatively to the sites. Since the distribution of the two classes is very unbalanced, the regularization also tends in our experiences to reduce the number of cracks detected. To overcome both issues, we introduce a correction of the mean value that is an affine function with respect to the line number, and set $\sigma_c = 1$ for simplicity.

In order to study the benefit of our approach regardless the site decomposition, we propose a “perfectly shaped” set of superpixels generated from the dilated ground truth. Then, for result derivation using actual superpixels, we have considered ETPS superpixels (Yao et al. [2015]) after image smoothing with median filtering with a square window of size (3×3) . Indeed, as mentioned in Section 4.3.1, ETPS superpixels exhibit good compactness (so that they can be efficiently modeled by their centroid) and regularity in size while at the same time allowing the grouping of crack pixels into thin superpixels.

Concerning the construction of anisotropic neighborhoods, the parameters are chosen so that there are 6 neighbors per site in average. For shape-based neighborhood, this boils down to setting the ellipse’s area to 7 times the mean area of a site. With cardinal-based neighborhood, we set $K = 4$ and $\beta' = 5 \times 10^{-3}$ in Equation (3.21). Finally for target-based neighborhood, in Equation (3.17), $\beta = 5 \times 10^{-3} \times \sqrt{\frac{A_V}{\pi}}$ where A_V is the area of the shape-based neighborhood ellipse (Equation (3.16)). Notice, these parameters are related to the relative scale of the thin structures with respect to the site size, and depend on each other.

5.3 Numerical experiments

5.3.1 Guidance map estimation

Before the evaluation of the performance of anisotropic regularization with respect to classic segmentation, we propose to compare visually the approaches proposed in terms of guidance map construction: Energy-based and TV. For doing so, Figure 5.1 provides a visualization of the orientations and saliency of the guidance map. These guidance maps are produced from the blind segmentation shown on the top left image, computed with ground truth based superpixels, the ground truth being displayed in the top right image. In the middle of the first row, the circle represents the legend for colormap used for guidance map. For energy-based guidance map, we note that the used neighborhood is the dictionary-based neighborhood having ellipses elements.

The visual comparison of the two methods shows us that energy-based model is very sensitive to the presence of FN detections, since it allows isotropic areas in the top left crack, for instance, and is also sensitive to FP detections, as seen in the middle of the image, contrary to TV. Additionally, while the orientation is quite well estimated far from the direct surroundings of a crack, our experiments show that it is not the case in the areas that are close to thin structures. This drawback is probably due to the fact that the elliptic shapes do not fit easily into the cracks. An interesting perspective could be to implement such guidance map with path-based dictionary neighborhoods. To cope with these errors, we can try better fitting with the regularization term β , but the latter point seems compromising, even when increasing β 's value, as seen for instance with the two blue stripes representing horizontal neighborhoods that should not exist in the top right corner for $\beta = 0.5$.

TV-based guidance map in contrast seems to produce smooth and robust maps, even for low values of σ_T . We tested various initialization options, for instance the use of stick tensors oriented from the shapes of the superpixels, but no improvement was brought compared to the simple ball tensor initialization that we finally used.

As energy-based guidance map appear too poor for allowing fine segmentation, we focus on TV guidance map in the next experiments, i.e. the qualitative comparison of the best segmentation obtained with respect to the considered measure (introduced in the next subsection), varying the values of α and β .

5.3.2 Qualitative and quantitative evaluation

Since our ground truth can be composed of 1 pixel width objects, in order to distinguish between slight location errors and non-detection of some parts of the cracks, we compute the *F-measure* (FM) at scale $\epsilon = 2$, based on the number of *True Positive* (TP), FP and FN, as in [Aldea and Le Hégarat-Masclé \[2015\]](#); [Vandoni et al. \[2016\]](#). In addition, the crack region and the non-crack area being highly unbalanced (in favor to the non-crack area), we use a high value of $\gamma = 5$ in FM to increase sensitivity to FN with respect to FP:

$$\text{FM}(\gamma) = \left(1 + \frac{\gamma^2 \text{FN}}{(1 + \gamma^2) \text{TP}} + \frac{\text{FP}}{(1 + \gamma^2) \text{TP}} \right)^{-1} \in [0, 1]. \quad (5.2)$$

For each scene, we compute a segmentation varying α values in $[1/64, 1]$, and σ_T values in $[0, 2]$, and finally select the best result with respect to FM.

The results at pixel level are presented in Fig. 5.2 and those at superpixel level in Figure 5.3, respectively. For each image, that corresponds to a set of parameters including the type of superpixels and the type of neighborhood, we select the best result, according to the FM criterion, among the results obtained varying the parameters σ (the scale parameter of tensor voting) and α (the regularization parameter in Equation (2.3)).

At pixel level, Figure 5.2 illustrates the clear improvement of the quality of the results with the use of anisotropic neighborhoods. In the first image of crack, anisotropic regularization allows for enhancing the continuity of the detected cracks, even if some small gaps still fragment it. In the

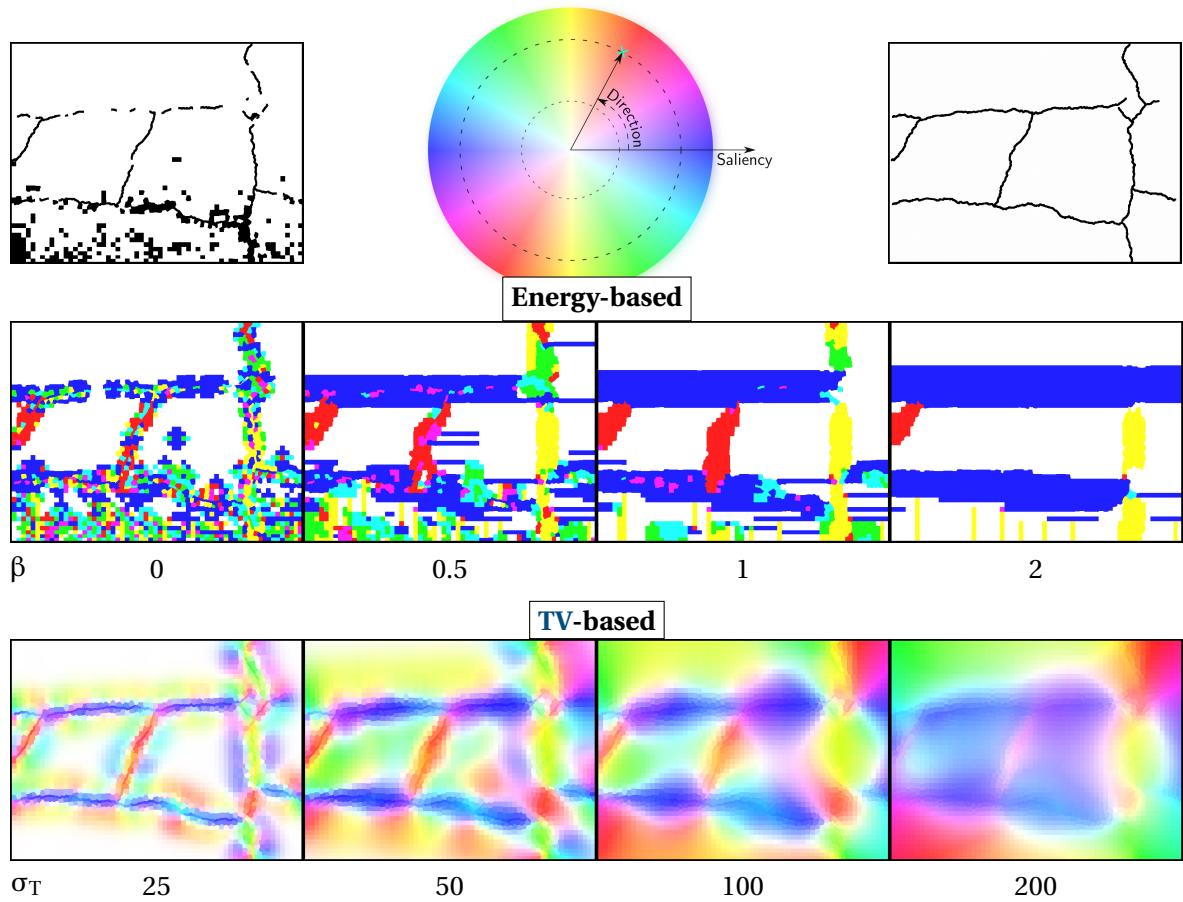


Figure 5.1 – Energy-based and TV-based guidance map comparison. These guidance maps are computed from the blind segmentation displayed in the image on the top left corner. The top right corner image is the ground truth, from which are computed the superpixels, and the image in the middle on the top is the legend of the colormap used for guidance maps. The two rows represent energy-based and TV-based guidance map, with various values of parameters β and σ_T , respectively. Visually, the TV-based guidance map seems to align correctly with the thin structures, even for low values of σ_T , and to be relatively robust to the presence of FN, while the energy-based model is very sensitive to them, and must be regularized, however, crucial information of orientation are quickly lost.

| Original image | Ground truth | Blind | Isotropic | Shape |
|----------------|--------------|-----------|-----------|-----------|
| | | 58.80 | 88.82 | 91.97 |
| | | 99.53 | 98.98 | 99.81 |

Figure 5.2 – Evaluation performance against ground truth at pixel level for a crack image (top row) and a simulated one (bottom row). The three last columns are segmentations without regularization (“blind”) and with regularization (isotropic with 4-connectivity or **shape**-based anisotropic neighborhoods with ellipses). For each image, in both regularized cases, the results achieving the largest FM with respect to tensor voting scale σ_T and regularization parameter α , are depicted. FM(γ) measurements are also provided in percents for $\gamma = 5$.

simulated image, the improvement appears quite significant: In particular, we notice the correct segmentation of the six discontinuities in the cracks, without loss of precision on more complex shapes.

However, at superpixel level (Figure 5.3), while exhibiting better blind results (than at pixel level) thanks to the noise filtering within superpixels, anisotropic neighborhoods seem to suffer of the difficulty to derive a correct neighborhood V even with a correct estimation of its orientation (see last column). Anyway, our experiments reveal that even if we are far from “optimal” neighborhood performance, path-based neighborhoods outperform the shape-based ones. Unfortunately, the anisotropic approach benefits exhibited in Figure 5.3 do not seem to improve the segmentation of the crack image in a so significant way: Path-based approach outperforms the other approaches only when superpixels are perfectly shaped, but are still sensitive to the degradation of the quality of the superpixels.

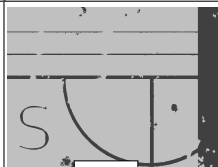
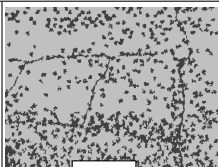
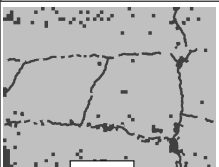
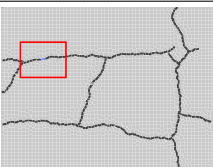

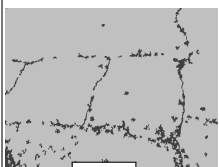

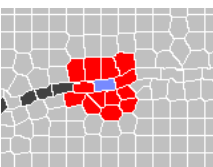
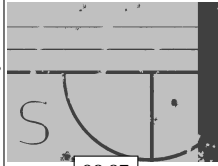
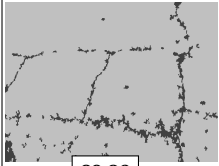
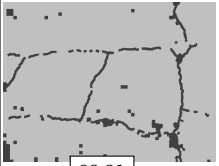
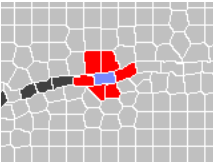
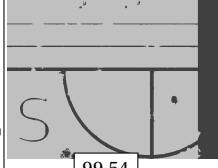
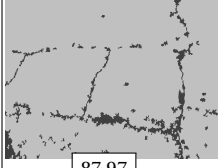
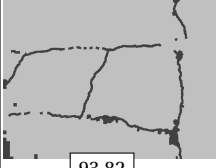
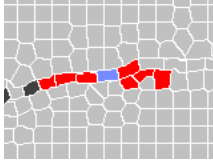
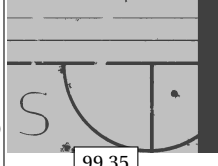
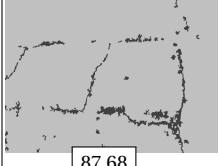
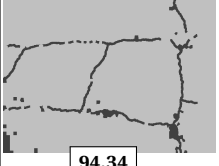
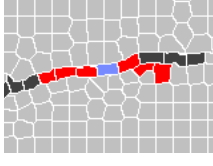
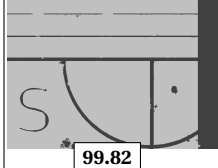
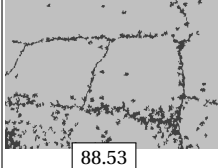
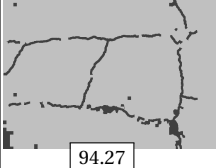
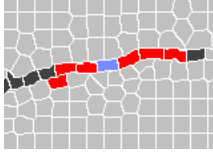
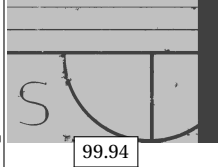
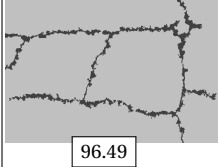
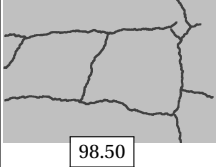

| | | Simulation | Crack image | | |
|-------------|-------------------|---|---|--|---|
| | | ETPS Yao et al. [2015] | “perfectly shaped” | | |
| Isotropic | Blind |  98.88 |  75.99 |  91.09 |  |
| | Disc Neighborhood |  98.84 |  88.78 |  92.90 |  |
| | Stawiasky |  98.87 |  89.36 |  92.81 |  |
| | Shape |  99.54 |  87.97 |  93.82 |  |
| Anisotropic | Target |  99.35 |  87.68 |  94.34 |  |
| | Cardinal |  99.82 |  88.53 |  94.27 |  |
| | Optimal |  99.94 |  96.49 |  98.50 |  |

Figure 5.3 – Evaluation of performance against ground truths at superpixel level for a crack image and a simulated one. For each isotropic neighborhood or TV-based anisotropic neighborhood, the segmentations achieving the largest $FM(\gamma)$ (for $\gamma = 5$) with respect to parameters σ and α are depicted. The last two columns correspond to the use of the “perfectly shaped” superpixel. The last column represents, for one site highlighted in blue, the sites that are in its neighborhood (in red), depending on the method used for constructing the latter one: Each row shows a different type of neighborhood, specified in header lines. The last row is a ground truth-based neighborhood for comparison purpose.

5.4 Conclusions

In this chapter, our experiments on a simple simulated image and an example of crack images allowed us to observe the behavior of our anisotropic approach for both energy based and TV based guidance maps, compared to isotropic neighborhoods. Visually, the second one seems to outperform the energy based guidance map, that lacks precision near the thin structures that we want to segment.

With the computation of regularized segmentations, we obtain mitigated results that are difficult to assess. While only a few results compete with isotropic regularization in these cases, this gives us a proof that the concept is promising, since it shows improvements in some areas that could not be possible with isotropic neighborhoods. However, the simulated image that shows the best improvements cannot be used as a solid evidence. Similarly with crack images, only ground-truth based superpixels provide slightly better results with anisotropic approaches, and the crack images remain problematic due to the very challenging task of producing suitable superpixels for thin structures in highly textured regions.

For these reasons, we complete our observations with a wider panel of scenes in a second application in the next chapter.

Chapter 6

Shape from focus (SFF)

Contents

| | |
|---|------------|
| 6.1 Problem and state of the art | 78 |
| 6.1.1 Principle | 78 |
| 6.1.2 Geometrical optics | 78 |
| 6.1.3 From physics to sharpness operators | 79 |
| 6.1.4 Depth derivation from sharpness | 79 |
| 6.1.5 SFF in variational framework | 81 |
| 6.2 Model application to SFF | 81 |
| 6.2.1 Data in SFF | 81 |
| Sharpness computation | 81 |
| Superpixel generation | 82 |
| Depth derivation | 82 |
| 6.2.2 Anisotropic regularization | 82 |
| 6.2.3 Thin structure estimation in SFF | 83 |
| Tensor voting | 83 |
| RORPO | 84 |
| 6.3 Proposed evaluation | 84 |
| 6.3.1 Depth error map | 84 |
| Depth error histogram | 85 |
| 6.3.2 Neighborhood quality estimation | 86 |
| Neighborhood orientation | 86 |
| Ground truth dynamic | 90 |
| Perfect neighborhood | 90 |
| Precision | 90 |
| IoU and threat score | 91 |
| 6.3.3 Global metrics | 93 |
| RMSE and PSNR | 93 |
| UQI | 93 |
| SSIM | 93 |
| 6.4 Experimental results | 94 |
| 6.4.1 Global segmentation results | 94 |
| Qualitative analysis | 94 |
| Successful scenes | 94 |
| Challenging scenes | 106 |
| 6.4.2 Sensitivity to the regularization parameter | 116 |
| 6.4.3 Result tables | 116 |
| 6.5 Conclusion | 127 |

6.1 Problem and state of the art

6.1.1 Principle

SFF is a very convenient method used for inferring the 3D shape of an object from a set of images with varying focus settings (Nayar and Nakagawa [1994]). SFF is an inverse problem and a passive optical method that presents the advantage of only requiring one fixed camera with a rather short depth of field and the ability to move this camera or to change the focal distance of the optical system (see Figure 6.1). SFF is therefore applicable in many real world applications including industrial inspection, micro manufacturing, robotic control, 3D model reconstruction, medical imaging systems and microscopy.

The reconstruction of the depth is performed thanks to the knowledge of the behavior of the Point Spread Function (PSF) of an optical system. The PSF is the function that associates to any point of an object, the image of this point on the sensor. This point image is either a point, when the point of the object is in the object focal plane, or a blurred spot, when the object is out of focus. The main idea behind SFF is therefore that the closer an object is to the object focal plane (i.e., the more it is focused), the more it appears sharp. Conversely, the farther an object is from this object focal plane, the more it appears blurred. Therefore, SFF makes use of a sharpness operator to find the depth where each point appears the more sharp, and reconstructs a depth image.

However, such an image is prone to noise and ambiguities since, in homogeneous or poorly textured area, the measured sharpness will be quite low and unreliable. Therefore, we formulate the problem in the variational framework in order to introduce some regularization terms. In this section, let us briefly recall the physics behind the principle of SFF and its resolution.

6.1.2 Geometrical optics

There are two main approaches to describe the formation of an image through an optical system. The first one, the *physical optics* approach, is based on electromagnetic wave theory and analyzes diffraction effects to compute the exact image formation model. However, in SFF, we rather stick to the second one, *geometrical optics* approach, that ignores diffraction effects to simplify the analysis. This approximation is justified by the short wavelength of light, that makes diffraction unnoticeable for our optical sensors that have a relatively limited resolution.

Figure 6.2 illustrates the geometrical notations involved in the image formation process. The optical system of the camera is assimilated to a thin lens within the framework of thin lens and paraxial approximation. The paraxial approximation deals with rays that form small angles with the optical axis of the system. This allows for setting $\sin(x) \approx x$, $\tan(x) \approx x$, and $\cos(x) \approx 1$, and establishing three rules:

- Any ray that enters parallel to the axis on one side of the lens proceeds towards the focal point F on the other side.
- Any ray that arrives at the lens after passing through the focal point on the front side, comes out parallel to the optical axis on the other side.

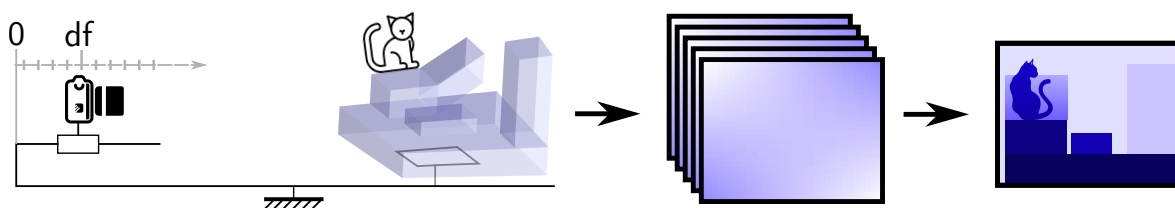


Figure 6.1 – An illustration of SFF. A camera (on the left) is moved along the optical axis by a distance df , or its focal setting are changed, while taking an array of pictures of the scene. Each of these pictures (middle) represents the scene with some objects blurred depending on their positioning. The principle of SFF is to reconstruct a depth image of the scene (represented on the right hand side image) from these pictures.

- Any ray that passes through the center of the lens will not change its direction.

From these rules the definition of object plane and image plane are derived, both normal to the optical axis, as well as the *Gauss lens law* or *thin lens equation* (6.1). Any ray crossing a point O in the object plane at distance (or depth) $s \in \mathbb{R}$ and passing through the thin lens then crosses a point P at distance $s' \in \mathbb{R}$ in the associated image plane such that:

$$\frac{1}{s} + \frac{1}{s'} = \frac{1}{f}, \quad (6.1)$$

where $f \in \mathbb{R}$ is the focal length of the lens. Then, if the image plane is also the sensor plane, the point O is said *perfectly* focused and its image is a single point P in the image.

But object planes and image planes go together as pairs of planes that are orthogonal to the optical axis and whose positioning respect the thin lens equation. Imaged scenes include points at variable depths. Therefore, the images of these points are localized in a multitude of image planes, whereas the sensor plane is unique for a given camera setting. For any non null shift between image and sensor planes, equal to δ , the energy received from the light that comes from the object through the lens is uniformly distributed over a circular patch on the sensor plane. The diameter $d' \in \mathbb{R}_{\geq 0}$ of this patch depends on the diameter $d \in \mathbb{R}_{> 0}$ of the surface of the lens that is crossed by light, which is the diameter of the diaphragm also called aperture of the optical system. This diameter is computed by using similar triangles as:

$$d' = \frac{\delta d}{s'}.$$

Such an effect is known as the defocusing effect: When an object does not belong to the plane that projects points to the sensor plane, it appears blurred. The resulting image on the sensor actually becomes similar to a convolution between the PSF and the input image, except that, since the PSF varies with depth of the image sources, the actual convolution involves a third dimension. In words, the more focused, the more sharp, and the more defocused, the more blurred are the objects representation in the acquired image.

6.1.3 From physics to sharpness operators

The idea behind **SFF** is to take series of pictures of the scene with varying focal settings and to discriminate sharp pixels in the set of images to recover depth (see Figure 6.3). For some applications, when the focal settings are fixed by design, another way of achieving the same result is to translate the object we want to measure in small steps towards the camera, or the camera may move towards the scene. In any cases, each setting corresponds to a specific depth for which the objects would be focused, and that is the reason why the set of 2D images is considered as a 3D volume.

Once the images have been acquired, the second step of **SFF** is to measure how focused is each point in the volume. This is done using a *sharpness measure* or *focus measure operator*. Typically, the sharpness measure quantifies the contrast in the surroundings of a pixel with the use of gradient or Laplacian operators, and averages the obtained values on a window centered on the pixel to produce the sharpness volume **Pertuz et al. [2013]**.

6.1.4 Depth derivation from sharpness

Initially the depth of each pixel of the 2D scene is computed as the one that maximizes the pixel's sharpness measure. **Nayar and Nakagawa [1994]** approximate the sharpness curve (that represents the sharpness values versus the focus parameter values) with a Gaussian model, and interpolate (along the optical axis) the three focus measures centered on the maximum sharpness value to allow for a better depth estimation. To reduce the sensitivity to noise, some authors do the sharpness curve interpolation by using quadratic, cubic or polynomial interpolation [**Moeller et al., 2015**] or Gaussian interpolation [**Ribal et al., 2018**].

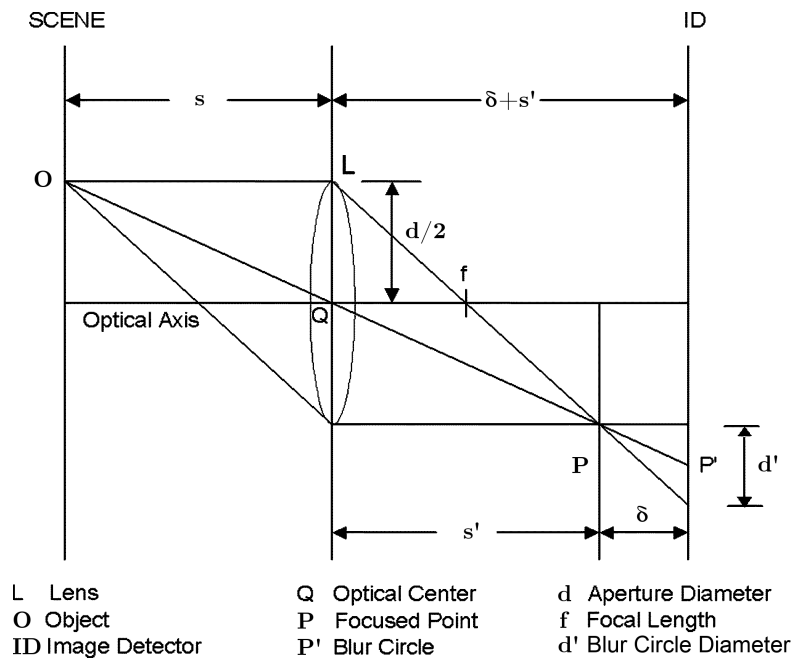


Figure 6.2 – The image formation scheme: In geometric optics, for a thin lens of focal f in the paraxial approximation, a point O in the object plane at distance s is associated to a point P in an image plane whose position s' is determined by Equation (6.1). With an aperture of diameter d , moving the sensor (or the object) away from the image plan (respectively the object plan) introduces blur: This process is called the defocusing effect. The diameter d' of the blurred image of point O is approximated by the point spread function derived from geometry. (Courtesy of Ahmad and Tae-Sun Choi [2005]).

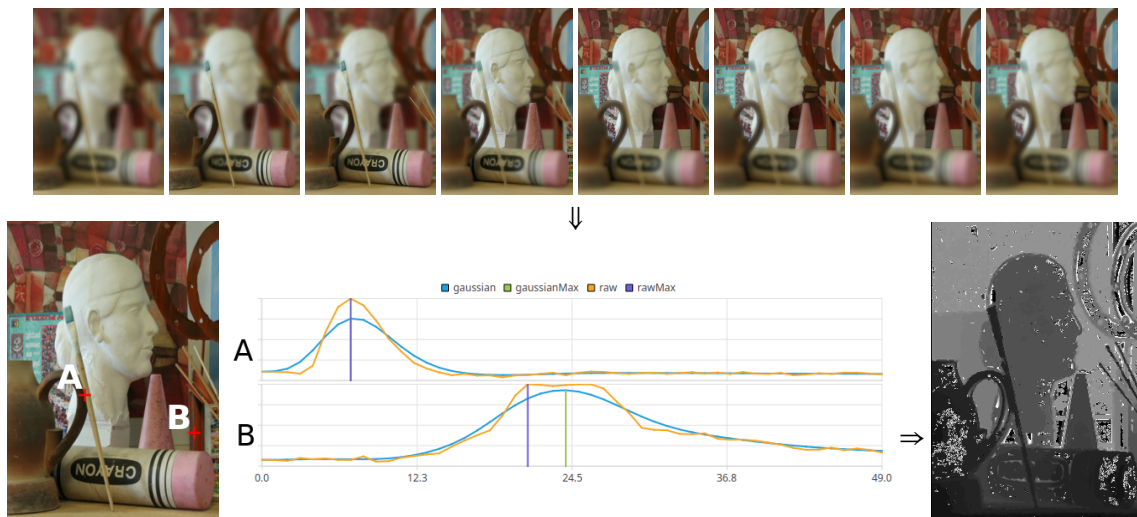


Figure 6.3 – Depth reconstruction in SFF, from the images acquired for scene *Art* (top row). For two sites (A) and (B), marked in the all-in-focus image (on the left), the sharpness curves are represented in the middle image. From the maximum of sharpness, a blind depth map (right image on the second row) is derived. The multiple curves illustrate options for computing the depth, in orange, from the raw sharpness data, and in blue, from gaussian interpolation [Ribal et al., 2018].

The result of such an estimation is however obviously limited in homogeneous regions: Because sharpness is supposed to be null in such areas, even a small Gaussian noise on the grey level values of the set of images can produce a Gaussian noise on the sharpness curve. Since each focus value is prone to the same Gaussian noise, this produces an uniform noise of maximal intensity on depth map. Without further improvements, the locally estimated depth map appears to be inaccurate in such areas excepted in textured regions and on the edges of the objects.

6.1.5 SFF in variational framework

To overcome previously mentioned limits, some authors have considered SFF in the variational framework, such as Gaganov and Ignatenko [2009]; Moeller et al. [2015]; Ribal et al. [2018]. With regularization, the information extracted in the scene areas where SFF is reliable, such as in objects details, contours, is propagated from neighbors to ambiguous areas, such as homogeneous, overexposed or underexposed regions. However, the presence of thin structures induces a need for anisotropic regularization.

6.2 Model application to SFF

6.2.1 Data in SFF

We test our approach on some scenes extracted from the Middlebury College dataset of 2005 and 2006¹, with the third-sized images. For each scene, we use a ground truth and an all-in-focus image to generate a set of simulated blurred images thanks to the defocusing algorithm of Pertuz et al. [2013]². We note that each image is a plane for which a basis is $(\mathbf{e}_0, \mathbf{e}_1)$ and a normal is \mathbf{e}_2 . The multiple images correspond to different focal object plane depths along the axis \mathbf{e}_2 . For simplicity and readability, we consider taking images at regular steps and set this step to be the unit. The maximum depth on this axis is therefore a multiple of the unit denoted by $\Delta_h \in \mathbb{N}$, but images taken at irregular steps could be considered without loss of generality as well.

Then, the set of defocused images is assumed to be the only input data available, and we recover depth from it based on the following steps: First, we compute the sharpness in each pixel independently to derive the blind depth map. In the case of sites that are superpixels, we can then compute the superpixels from an approximated all-in-focus image. Finally (as described in Section 6.2.2), we instantiate our anisotropic regularization through the definition of an energy and the adaptation of our guidance map and neighborhood construction to SFF.

Sharpness computation

Indeed, sharpness information cannot be computed at superpixel level, since sharp information is lost by the phenomenon of averaging the color of the pixels into superpixels. Therefore, we compute the sharpness at pixel level beforehand, and if needed, the sharpness at superpixel level would be derived from the mean sharpness values of the pixels that compose it.

The sharpness operator we use is the Summed Modified LAPlacian (SMLAP) introduced in Pertuz et al. [2013]. For each pixel $p \in \mathcal{P}$, it is defined as

$$\text{SMLAP}(p) = \mathbf{f}(p) = \sum_{\substack{q \in \Omega(p) \\ q=(i,j,k)}} \left(\left\| \frac{\partial I(i,j,k)}{\partial \mathbf{e}_0} \right\|_1 + \left\| \frac{\partial I(i,j,k)}{\partial \mathbf{e}_1} \right\|_1 \right) \in \mathbb{R}_{\geq 0}$$

where $I(i,j,k) \in \mathbb{F}$ is the intensity of the image at pixel $q = (i,j,k) \in \mathcal{P}$, where k is the depth, and $\Omega(p)$ is a window centered on the pixel $p \in \mathcal{P}$. In our case, $\Omega(p)$ is a flat squared window of size $7 \times 7 \times 1$ but other choices are possible.

¹Scharstein and Pal [2007], dataset available currently on <https://vision.middlebury.edu/stereo/data/>

²The defocus simulation algorithm is currently available as a Matlab source on MathWorks file exchange at <https://fr.mathworks.com/matlabcentral/fileexchange/55095-defocus-simulation>.

From the maximum of sharpness, we can estimate the all-in-focus image. For each pair of coordinates $(i, j) \in \mathbb{N}^2$:

$$\text{AF}_I(i, j) = I(i, j, k), \text{ with } k \in \underset{k \in [0, \Delta_h]}{\operatorname{argmax}}(\mathbf{f}(i, j, k)).$$

If needed, this image allows us to compute superpixels that have a good sensitivity to the sharp edges of the scene, everywhere in the 2D space, since it picks for each pair of coordinates the pixel that is the sharpest, that is to say that has the most contrast with its neighbors. The chance of constructing superpixels on blurred edges of the objects is minimized this way.

Superpixel generation

Since SFF data is composed of a lot of frames represented in 3D, we denote by \mathcal{S}_3 the set of the sites in 3D. For computing the sites in the case of superpixels, we first compute them on the 2D image named all-in-focus image AF_I , and we obtain the set of sites in 2D \mathcal{S} .

Then, the sites of \mathcal{S}_3 are constructed by replicating the sites of \mathcal{S} along the third dimension, as depicted in Equation (2.6). For later use, we will denote the set of sites that correspond to the 2D coordinates $(i, j) \in \mathbb{R}^2$ of a given site $s \in \mathcal{S}$ by:

$$\mathcal{H}_{(i,j)} = \mathcal{H}_s = \{p \in \mathcal{S}_3 \mid \vec{sp} \propto \mathbf{e}_2\}.$$

Depth derivation

Having defined the sites set, the sharpness of a site is defined in an ad hoc way as the mean of the sharpness of its pixels. $\forall s \in \mathcal{S}_3$:

$$\mathbf{f}(s) = \frac{1}{\#\{p\}_{p \in s}} \sum_{p \in s} \mathbf{f}(p) \in \mathbb{R}_{\geq 0}.$$

Then, the depth is naturally computed as the depth of the maximum of sharpness at site level. We define the blind depth map as $\hat{\mathbf{u}} = (\hat{u}_s)_{s \in \mathcal{S}}$:

$$\begin{aligned} \hat{u}_s &= |\vec{st}|, \\ \text{with } t &\in \underset{p \in \mathcal{H}_s}{\operatorname{argmax}}(\mathbf{f}(p)), \\ &\forall s \in \mathcal{S}. \end{aligned}$$

Naturally, this depth map may be noisy and sensitive to the low sharpness profile of homogeneous regions of the scene, which we cope with thanks to our anisotropic neighborhood based regularization. When $\exists s \in \mathcal{S} / \max_{t \in \mathcal{H}_s}(\mathbf{f}_t) < \kappa_f$, where κ_f is a small positive real, we found that the depth estimate \hat{u}_s may be erroneous and could lead to a constant bias. For that reason, we prefer, in such a case, to select a uniform random blind initialization \hat{u}_s . In our experiments, we set $\kappa_f = 2$.

6.2.2 Anisotropic regularization

We implement the functional presented in Equation (2.3), with the terms E_1 and E_2 instantiated respectively with quadratic distance to the blind estimate \hat{u}_s and total variation Equation (2.5), $\forall \mathbf{u} \in \mathbb{N}^{\mathcal{S}}$, and for $V \in \mathbb{V}$:

$$F(\mathbf{u}, V) = \sum_{s \in \mathcal{S}} W_s (u_s - \hat{u}_s)^2 + \alpha \sum_{(s,t) \in \mathcal{N}} W_{st} |u_s - u_t|,$$

where

$$W_s \propto \left(\frac{\Delta_h \times \left(\max_{t \in \mathcal{H}_s}(\mathbf{f}(t)) - \min_{t \in \mathcal{H}_s}(\mathbf{f}(t)) \right)}{\sum_{p \in \mathcal{H}_s} \left(\mathbf{f}(p) - \min_{t \in \mathcal{H}_s}(\mathbf{f}(t)) \right) + \epsilon} \right), \quad (6.2)$$

where $\alpha \in \mathbb{R}_{>0}$ is an hyper parameter, $\epsilon \approx 0$ and W_{st} is the weighting function depending on the neighborhood V .

With this weighting term W_s , the fidelity to the data term is decreased when the sharpness profile is homogeneous, or when it presents a very low dynamic. On the contrary, the areas with a sharpness profile with a localized high response have high values of W_s reflecting the belief that they are trustful. We note that compared to our binary implementation, the data used for E_1 is not the radiometric information of the input image, but the blind depth derived from the sharpness profile.

6.2.3 Thin structure estimation in SFF

In Chapter 3, we have introduced both the construction of a guidance map from thin structures estimation and the construction of anisotropic neighborhoods from the latter guidance map. However, specific aspects linked to the implementation have been overlooked in this general presentation and must be detailed in this section. First, we will introduce our implementation for the TV operator, and then for RORPO.

Note that when we perform segmentation at pixel level, since the number of pixels is very high for TV and RORPO computation, we nevertheless compute the guidance map at superpixel level. The input data used for both approaches is the sharpness profile $\mathbf{f} \in \mathbb{R}_{\geq 0}^{\mathcal{S}_3}$.

Tensor voting

Initialization In SFF, for each pair of coordinates (i, j) corresponding to a location in $(\mathbf{e}_0, \mathbf{e}_1)$ plane, the token selected to initiate the vote are chosen among the local maxima of sharpness in $\mathcal{H}_{(i,j)}$. However, since the sharpness profile is likely to be noisy, one faces multiple close maxima possibly inducing as many initial tensor that will reinforce artificially. To cope with this, we introduce a non-maximum suppression (specifically, we keep only one maximum per continuous interval of focus values associated to sharpness values greater than 80% of the maximum of sharpness value in $\mathcal{H}_{(i,j)}$). This way, we ensure that initial voters are all separated by a local minimum which value is under 80% of the global maximum.

This initialization provides a tensor map that is sparse in 3D, but dense in 2D.

Voting steps In our implementation of TV for SFF, since the number of pairs of sites in $(\mathcal{S}_3 \times \mathcal{S}_3)$ is very large, we restrict the vote for the orientation estimation to the initial set of tokens. This allows for reducing the computational burden by removing the dense voting step, at the risk of a loss of accuracy when the initial depth estimations (and thus voters) are erroneous. However, as stressed, the voting remains dense when projected into 2D space.

3D to 2D guidance map derivation Similarly to RORPO, TV allows for computing a set of tensors that are defined in \mathbb{R}^3 , instead of \mathbb{R}^2 . However, the choice of the tensor that best describes the orientations of thin structures for any site $s \in \mathcal{S}$ is not trivial. Having tested different possibilities such as performing dense voting and various estimation methods, either based on stick, ball or plate maximum saliency or relative saliency maximum detection, we finally found that the more convincing results are obtained when only considering the cumulated tensors (after voting) at the blind estimated depth $\hat{\mathbf{u}}$. Indeed, we have seen through our experiments that while this leads to irregularities when $\hat{\mathbf{u}}$ is noisy, this also allows for gaps in the orientations estimated on the edges of the structures of the images, which could be beneficial.

The method for extracting the guidance map from the 2D map of tensors is then to project the tensor into 2D space to derive the major eigenvector's orientation. Our tensors are 3×3 matrices of contravariant coordinates that associate to each vector of the plane a second vector. The projection of the tensor \mathbb{T} in $(\mathbf{e}_0, \mathbf{e}_1)$ corresponds to null coefficients in the third line and third column of the tensor.

From this projected tensor, we extract the major orientation, and two eigenvectors $\lambda_0, \lambda_1 \in \mathbb{R}_{\geq 0}^2$, as explained in Section 3.2.2.

RORPO

Just as with TV, the RORPO algorithm uses the sharpness profile in Equation (3.13). Then the data is dilated, and the grey level openings realized. One important point is that we only consider 2D neighborhoods, which means that the path can only spread at fixed depth. This is a parameter of the algorithm that we fixed empirically after achieving better results than with 3D neighborhoods allowing paths between sites at different depth. Another parameter that we have tested is the normalization of the sharpness profiles along each set \mathcal{H}_s , so that the effect of the local texture of the scene would be minimized with respect to the effect of being localized at the more focused depth. However, this does not yield good results, and we limited ourselves to the use of the dilated sharpness profile without normalization.

3D to 2D guidance map derivation After computation of the RORPO, the guidance map \mathbf{g} in Equation (3.15) is computed for every site $s \in \mathcal{S}_3$ in \mathbb{R}^3 , while we expect the neighborhood to be computed in 2D. To get 2D values, we average the directions in any given set \mathcal{H}_s with the same technique than in Equation (3.14), $\forall s \in \mathcal{S}$:

$$\mathbf{g}_s = \left(\sum_{t \in \mathcal{H}_s} |\mathbf{g}_t| \exp(2i \arg(\mathbf{g}_t)) \right)^{1/2}, \quad (6.3)$$

where $\arg(c)$ denotes the argument of any complex number $c \in \mathbb{C}$.

6.3 Proposed evaluation

The purpose of this chapter is to evaluate the benefits of our approach on the application of SFF compared against isotropic neighborhoods. Since isotropic neighborhood using discs of fixed size had worse performances than Stawiaski and Decencière [2011] in Chapter 5, we removed it from the comparison for readability. In this section, we introduce evaluation methods for our application in which, for recall, the labels are ordered. We can therefore use measures that are often used in image restoration. We do these measures at pixel level. When the sites are superpixels, we note that there is a maximum value of an optimal segmentation (or depth map), that can also be measured. All along this section, we illustrate these quality observations with images from a sample of our dataset, *Brush2*, that is a toy example.

Let $\mathbf{x} = (x_p)_{p \in \mathcal{P}} \in \mathcal{C}^{\mathcal{P}}$ and $\mathbf{y} = (y_p)_{p \in \mathcal{P}} \in \mathcal{C}^{\mathcal{P}}$ be respectively the ground truth image and the segmentation image at pixel level. In the rest of this manuscript, however, please note that the segmentation obtained is $\mathbf{u} = \mathbf{y}$. We show the all-in-focus image, the ground truth and an example of segmentation for *Brush2* in Figure 6.4 We define such ground truth at site level in the next subsection.

At superpixel level, the ground truth $\mathbf{y} = (y_s)_{s \in \mathcal{S}} \in \mathcal{C}^{\mathcal{S}}$ is computed as the mean of the depths of the sites that constitute it, $\forall p \in \mathcal{S}$:

$$y_s = \sum_{p \in \mathcal{S}} \frac{y_p}{\#\{p\}_{p \in s}}. \quad (6.4)$$

6.3.1 Depth error map

A first qualitative and intuitive evaluation is provided by the error map with error defined $\forall p \in \mathcal{S}$ as $\epsilon_m(p) = |x_p - y_p|$. The visualization of this error as a heat map with colors allows us to represent the error of segmentation more visually and to locate erroneous areas. In practice, the presence



Figure 6.4 – Brush sample: All-in-focus image (left), ground truth (middle) and an example of segmentation (right). In the ground truth and depth map, the darkest areas are the closest areas, while brighter areas are further.

of some outliers can affect the error map dynamic. Since we are less interested by these very local errors than by more significant and spatially expended errors, the presented error maps are bounded to two fifth of the maximum dynamic of the segmentation:

$$\epsilon_m(p) = \min \left(|x_p - y_p|, \frac{2\Delta_h}{5} \right).$$

Figure 6.5 shows an example of such bounded error map. Besides, due to the important size of the set of parameters, an exhaustive visual analysis of the experiences would be intractable. In this presentation, results are thus displayed varying only the two main parameters: The neighborhood type, and the regularization parameter. For example, in the case of our toy example *Brush2*, Figure 6.6 and Figure 6.7 allow for comparison between the results obtained with our six different neighborhood implementations (cardinal-based, target-based, shape-based, for both TV and RORPO-based guidance map) and results provided using the perfect neighborhood and Stawiaski and Decenci re [2011]’s neighborhood.

Figure 6.7 showing the depth error allows us to better visualize the difficult areas, i.e. the areas where the errors are mainly located. As expected, the errors mostly concentrate around the sharp edges of the brushes, that are sharp edges of the ground truth.

Depth error histogram

To have additional information, we plot the histogram of errors, and derive the corresponding boxplots. The histogram displays the distribution of the depth error. We plot it at logarithmic scale, since some bins of the histogram have very high values. However, since this prevents the direct comparison of the neighborhood performances, we also show the corresponding boxplots.

A boxplot allows us to visualize the location of the mean value, of the first and last quartile, and of the first and last decile of a distribution. In Figure 6.8, the positioning of the mean is marked with a circle with a dot, the first and last quartile are delimited by the thick vertical line, and the delimitation of the last decile is shown by a thin vertical line. The outliers, data points that are farther than 1.5 times the interquartile range from the median, are also represented as small circles.

The histograms and boxplot of Figure 6.8 correspond to the best estimation of depth maps among those shown in Figure 6.6 and Figure 6.7.

We compare the neighborhood performance from the boxplots, that visually allow us to distinguish three groups. Firstly, the group of *perfect* and cardinal-based neighborhood exhibit the best results, with a small mean value and a small dispersion until the last decile, and also a very small amount of outliers, nevertheless exhibiting quite small depth errors. Secondly, the target-based neighborhood have a higher dispersion of outliers. Thirdly, the last group gathers shape-based neighborhoods and Stawiaski and Decenci re [2011]’s neighborhood that have a much wider dispersion according to the value of the ninth decile.

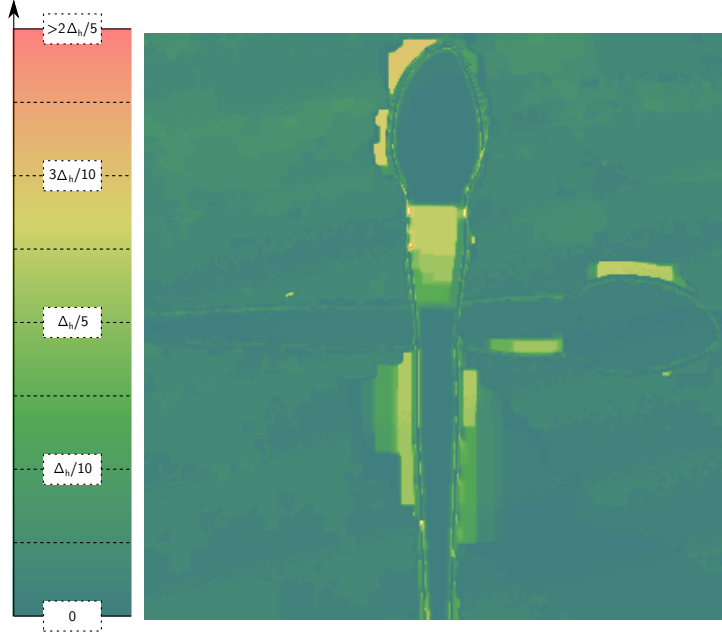


Figure 6.5 – Depth error map for the depth map in Figure 6.4 on the right hand side, with its legend on the left hand side. The color map ranges from blue (null error) to red (error higher than two fifth of the depth span). In this example, horizontal brush is quite well detected, as is the background, but the depth of the areas around the bristles of the vertical brush and the depth in the upper part of the handle of the vertical brush are badly estimated.

These plots allow for non-spatial numerical evaluation, thus being very complementary to the quality analysis of the error maps.

6.3.2 Neighborhood quality estimation

Knowing the localization of the errors gives important hints about the challenges that our anisotropic neighborhoods may help to overcome. To evaluate the quality of such neighborhoods, let us introduce indicators to visualize their performance.

We can represent the neighborhood orientations computed for each site with the guidance map in a single image. However in SFF the ground truth orientation of the thin structures is not known. Thus, the quality of the orientations will mainly be evaluated in a qualitative way. Moreover, the quality of the neighborhood itself is hard to appreciate, mainly since there is no such thing as a *perfect* neighborhood that would be the unique solution for thin structure preservation. For completing our analysis, we thus define several specific indicators in that section, after a short description of the orientation map.

Neighborhood orientation

The orientation map represents saliency and direction information extracted from the guidance map, computed by TV (Equation (3.12)) or by RORPO (Equation (3.15)), the saliency being the norm of the vector \mathbf{g}_s , $\forall s \in \mathcal{S}$, and the orientation being the angle between \mathbf{g}_s and the axis \mathbf{e}_0 , in the plane of the image.

For representing both information types, in Figure 6.9, we use a color representation, such that the saturation and the hue encode respectively the saliency and the orientation. Then, the orientation map is expected to be relatively smooth while following the sharp edges of the objects and aligning with the thin structures, to improve their segmentation.

However, this qualitative hint is not enough for appreciating the quality of the neighborhood construction that results from each guidance map, therefore we also introduce two criteria that we can also display as colormaps.

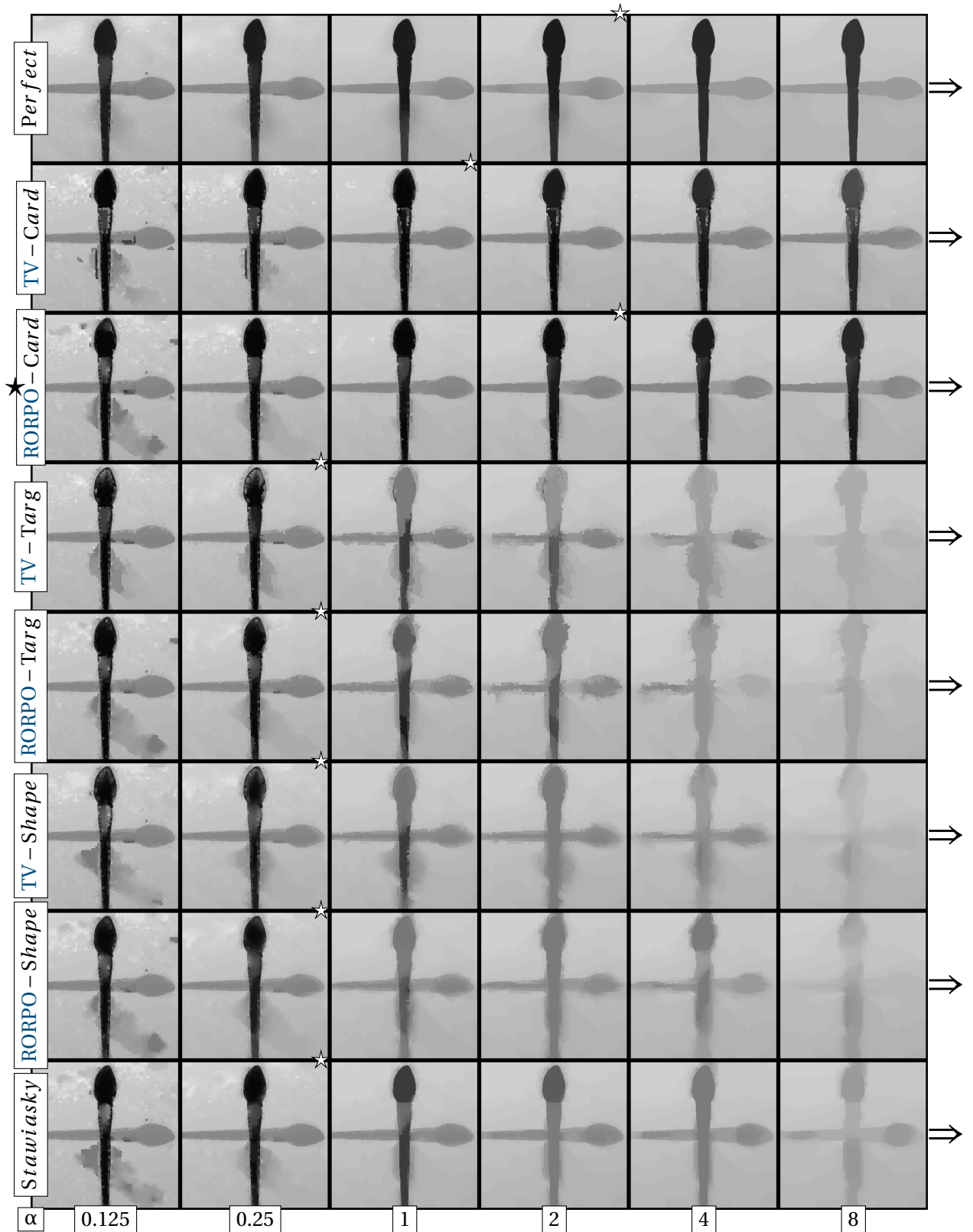


Figure 6.6 – Segmentation obtained for the scene *Brush2*, for different neighborhood strategies and different values of α (Equation (2.3)), which are indicated at the bottom. The arrows indicate the direction of increasing α , from left to right. For each row, the neighborhood used is indicated at the left hand side, and the best depth map obtained (with respect to the [Peak Signal to Noise Ratio](#) among a given row) is marked with a star. Finally, among the best results, the neighborhood that gives the best result (excluding the theoretically perfect one) is also designed by a black star.

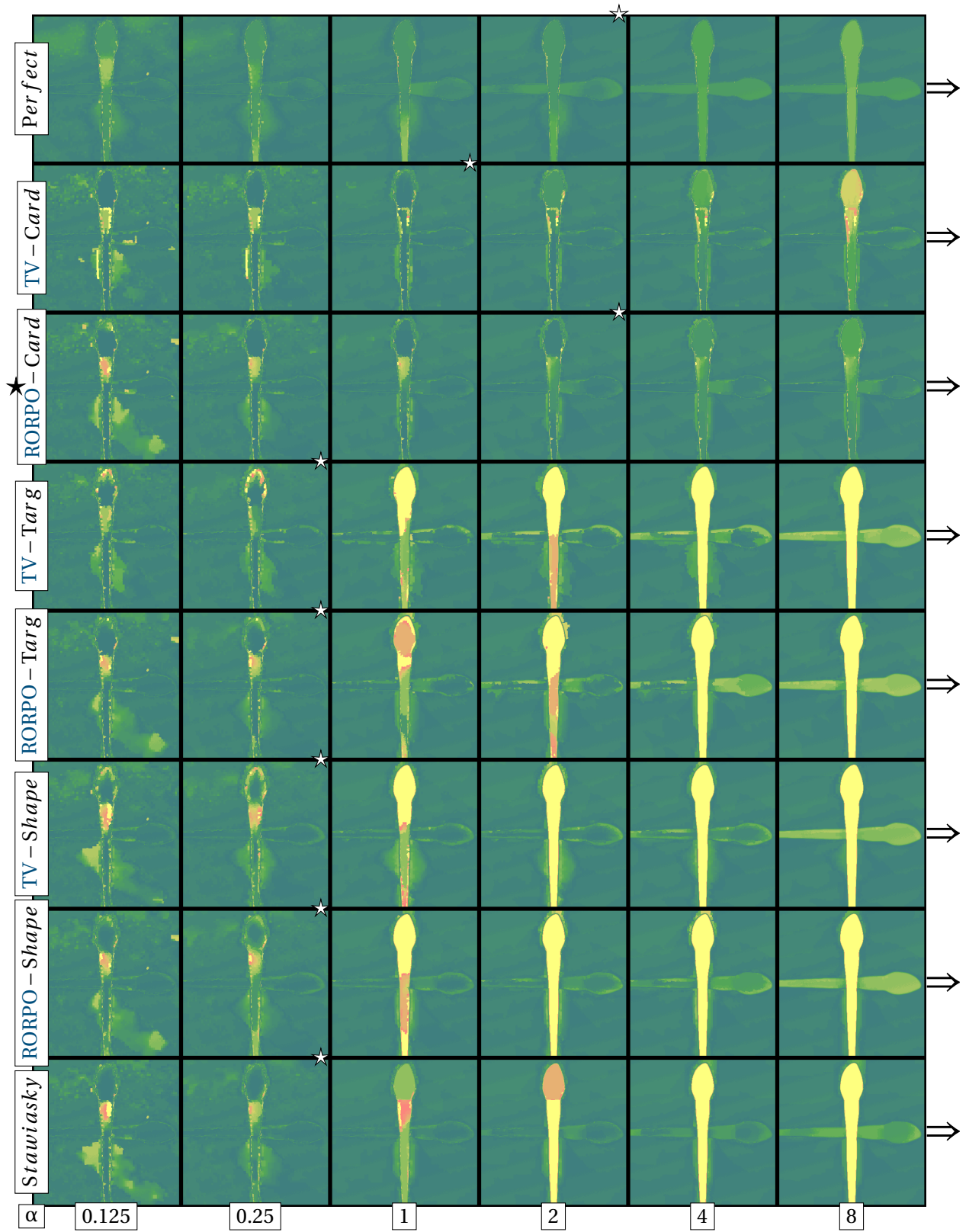


Figure 6.7 – Error maps obtained for the scene *Brush2*, for different neighborhood strategies named on the left hand side and different values of α (Equation (2.3)), which are indicated at the bottom. For each row, the best depth map obtained (with respect to the PSNR) row is indicated with a star. Finally, the neighborhood that gives the best result (excepted the theoretically perfect one) is designed by a black star. The color map the same as in Figure 6.5.

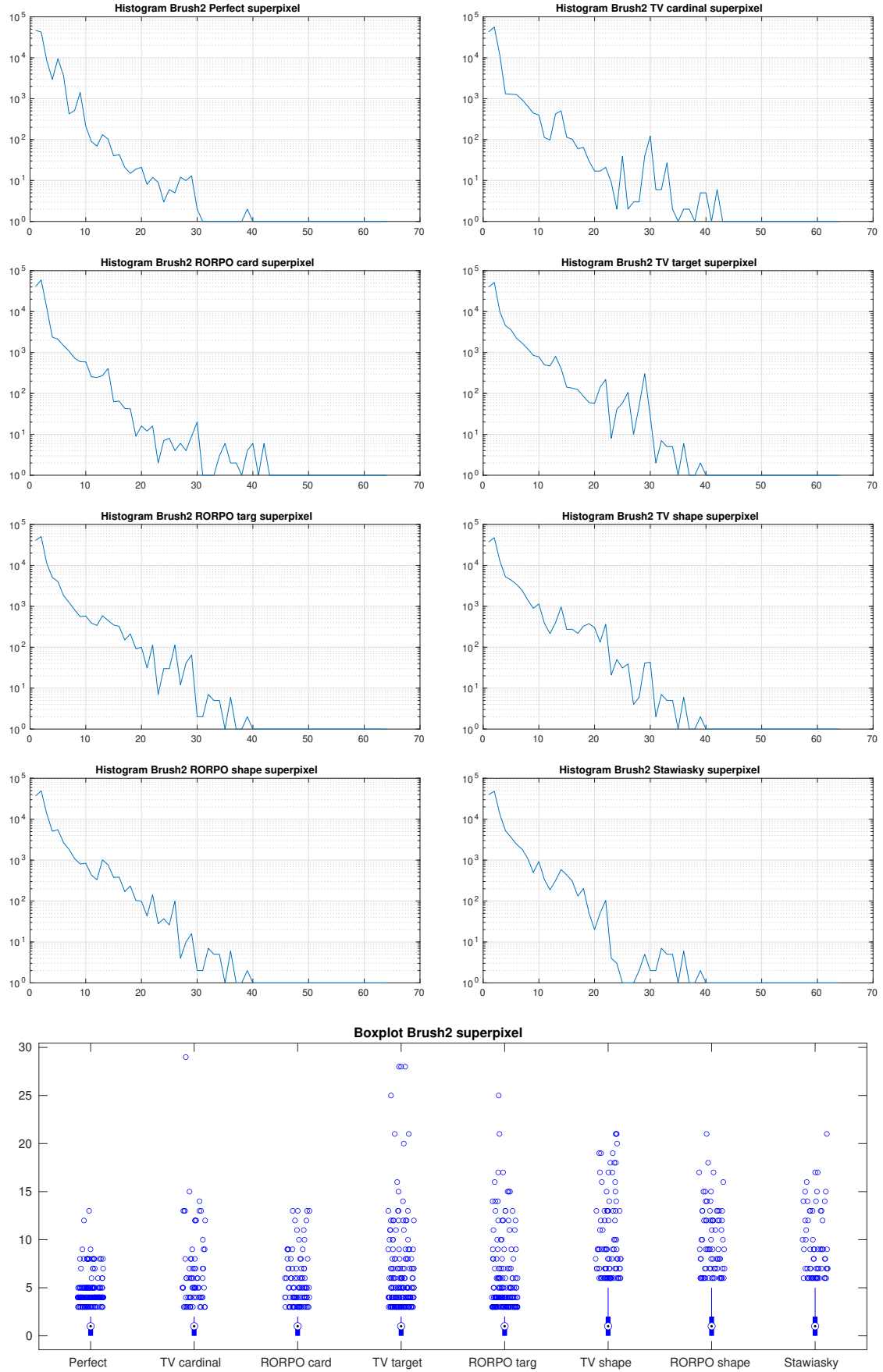


Figure 6.8 – *Brush2* depth error map histograms and boxplots corresponding to the best results of Figure 6.7. Because of the logarithmic scale, the histograms are difficult to compare visually. However, the boxplots in the last row give a simpler representation of the distribution, and allow us to compare and cluster the neighborhoods according to their performance.

Ground truth dynamic

Since we expect the regularization to smooth depths within the neighbors of a site, the dynamic of the ground truth within these neighbors is expected to be as low as possible. We therefore propose a criterion underscoring the configurations where the ground truth labels x_q of the neighbors of a site p are close to its label x_p , $\forall p \in \mathcal{S}$:

$$Q_V(p) = \max_{q \in V(p)} |x_q - x_p| \in [0, \Delta_h],$$

where $\mathbf{x} = (x_p)_p \in \mathcal{S}$ is the ground truth expressed at site level.

Note that such a criterion only works with ordered labels (as are depth values in our case). Figure 6.9 presents an example of this quality measure for the eight neighborhoods we consider, with reversed the dynamic in $[0, 255]$, so that dark area represent bad performance.

A major benefit of such an evaluation criterion is that it does not require any neighborhood ground truth. Another advantage is its straightforward interpretation in terms of neighborhood consistency measure. However, such a definition does not take into account the geometry of the neighborhood: For instance, a valid neighborhood for this criterion could assign scattered neighbors $\{p\}_{p \in V(s)} \subset \mathcal{S}$ to each site $s \in \mathcal{S}$ as long as their depth x_p in the ground truth is equal to the depth x_s of s . This can be seen either as an advantage in case of multiple solutions, or a disadvantage since we want to evaluate the neighborhood geometry.

Perfect neighborhood

In this subsection we propose a criterion based on *Precision* with respect to a *perfect* neighborhood V' . However, since we do not have a ground truth for thin structures and neighborhoods, we see in the next sections that the word *perfect* must not be taken literally as the only acceptable construction for a neighborhood.

We define an ideal neighborhood that performs well for the criterion involving ground truth dynamic, while at the same time limiting the neighbors of a site to the ones that located close to it. Typically, this neighborhood is implemented as a shape-based neighborhood with a disc of constant radius as unique shape, where neighbors of a site $s \in \mathcal{S}$ for which the ground truth depth difference with s is higher than a fixed threshold $D_V = (\Delta_h / 10) + 1$ are removed. We can write it as, $\forall s \in \mathcal{S}$:

$$V'(s) = \{p \in \mathcal{S} \text{ such that } |\vec{sp}| \leq \hat{R} \text{ and } |x_q - x_p| < D_V\},$$

where \hat{R} and D_V are respectively the radius of the disc for constraining geometrically the neighborhood, and the depth threshold for constraining the neighborhood. Additionally, elements that do not form a single connected component with s are removed from the neighbors of s , $\forall s \in \mathcal{S}$. We select a high radius $\hat{R} = \sqrt{(40 * \hat{A}_S) + 1}$, where \hat{A}_S is the mean area of the sites (that reduces to 1 if the sites are pixels), such that the set of neighbors of a site is the closest to what could be the union of all the sites that should be acceptable as neighbors. The perfect neighborhood V' therefore acts as an upper bound for V , and ideally, for any neighborhood V , $V(s)$ should be a subset of $V'(s)$.

Since the perfect neighborhood has a large radius, note that its performances with the criterion $Q_{V'}$ may be lower than the one of the other neighborhoods, despite being bounded by the value of the depth threshold. By comparison, the error Q_V for Stawiasky's neighborhood is much more localized, but not bounded.

For this reason, we also compute a quality map based on Precision.

Precision

This quality criterion ranges in $[0, 1]$, the higher the better, and is based on the computation of FP and TP values as follows, $\forall p \in \mathcal{S}$:

$$Prec_{V,V'}(p) = \frac{TP(p)}{TP(p) + FP(p)}, \quad (6.5)$$

| | $p \in V'(s)$ | $p \notin V'(s)$ |
|-----------------|---------------|------------------|
| $p \in V(s)$ | TP | FP |
| $p \notin V(s)$ | TN | FN |

Table 6.1 – Confusion matrix for counting the number of TP, TN, FP, FN for neighborhoods.

where a true positive is defined as a site p belonging to the neighbors of a site $s \in \mathcal{S}$ for both the neighborhoods V and V' . The FP are computed as sites belonging to V without belonging to V' . See Table 6.1 for visual explanation of TP, FP, *True Negative* (TN), FN in the case of neighborhoods. Precision, or positive predictive value does not include the count of FN and therefore corresponds better with our expectations: Neighbors of s for V should belong to the perfect neighborhood V' , but the reciprocal is not true. We illustrate this criterion on the eight neighborhoods in Figure 6.9.

We can see that the shape-based neighborhood exhibit the worst performances near the sharp edges of the objects, and that the regions of bad performances seem to correspond to the regions where the depth estimation seem to be erroneous.

We also draw the readers attention on the fact that the perfect neighborhood shows an imperfect quality map for $Prec_{V',V'}$ since a site sometimes has no neighbors, which results in Equation (6.5) being mathematically unstable. This happens when the sites are isolated by a sharp difference of depth from the adjacent sites. We added in the definition a small coefficient in the denominator to cope with this phenomena, which explains the darker sites in the quality map. For these sites, since the count of TP is always null, the performances of the other neighborhoods are also impacted and can be ignored.

IoU and threat score

Precision is derived from Intersection over Union (IoU), initially considered for comparing geometrical shapes. In this subsection we show how IoU relates to the *threat score*, and how ignoring the FN lead us to use *Precision* instead.

The IoU for our neighborhood would be computed as follows:

$$IoU_{V,V'}(p) = \frac{\# \{V(p) \cap V'(p)\}}{\# \{V(p) \cup V'(p)\}}, \quad (6.6)$$

ranges in $[0, 1]$ and reaches its maximum when the sets of neighbors of p in $V, V' : \mathcal{S} \mapsto 2^{\mathcal{S}}$ are equal. If, in this division, we consider the sites as units instead of computing their areas, this formulation is very close to the *Threat Score* defined $\forall p \in \mathcal{S}$ as:

$$TS(p) = \frac{TP}{TP + FN + FP}, \quad (6.7)$$

where TP, FN, FP are defined in Table 6.1. To take into account the fact that the *perfect* neighborhood includes more neighbors than required, we implemented other representations of the quality of our neighborhood such as precision and a quality estimation based on the segmentation ground truth dynamic. Because the FN are taken into account, the *Threat Score* criterion is not suitable for our evaluation and we replaced it with *Precision*.

Now that we have introduced criteria that allows us to visually interpret the quality of neighborhood construction and depth reconstruction, we need quantitative metrics to corroborate our observations and draw conclusions over larger set of experiments, including more scenes of the dataset.

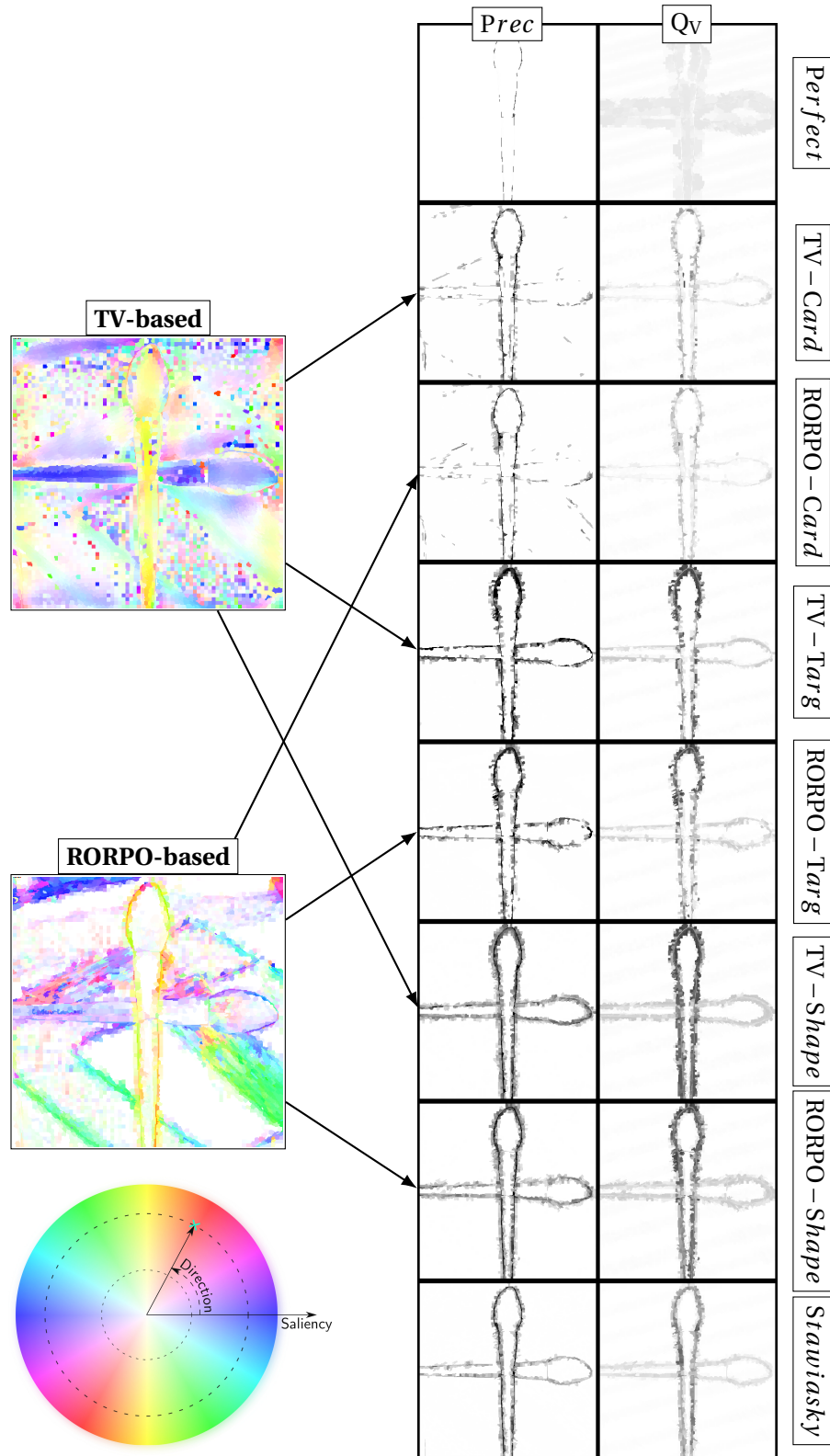


Figure 6.9 – On the left hand side, we present the neighborhood guidance map orientation and saliency, computed using the colormap which is represented below as a legend. Our two neighborhood qualitative criteria are displayed for each of the eight neighborhoods on the right hand side. The guidance map representation allows us to visualize the behavior of both TV and RORPO algorithm: While TV offers smooth orientations based on the location of the voters, RORPO guidance map highlights the thin areas of high sharpness values in the scene, may they be on the same depth or not. The quality criteria each show a specific nature of error on the neighborhood: The first column penalizes low precision with respect to the theoretical neighborhood, while the second column penalizes the high dynamic of the ground truth in the neighborhood.

6.3.3 Global metrics

Global metrics allow for a quantitative comparison of the depth reconstruction for any setting of the parameters on the whole dataset. We introduce four measures frequently used in image reconstruction, the [Root Mean Square Error \(RMSE\)](#), the [PSNR](#), the [Universal image Quality Index \(UQI\)](#) and the [Structural Similarity Index Measure \(SSIM\)](#). For computing these measures, $\mathbf{x} \in \mathcal{C}^{\mathcal{P}}$ and $\mathbf{y} \in \mathcal{C}^{\mathcal{P}}$ are respectively the ground truth and the segmentation (or depth map) at pixel level.

RMSE and PSNR

First of all is the well known RMSE, non negative, that has to be minimized:

$$\text{RMSE}(\mathbf{x}, \mathbf{y}) = \sqrt{\frac{\sum_{p \in \mathcal{P}} (x_p - y_p)^2}{\#\mathcal{P}}} \in [0, \Delta_h],$$

It is correlated to the PSNR, that has to be maximized:

$$\text{PSNR}(\mathbf{x}, \mathbf{y}) = 20 \log_{10} \left(\frac{\Delta_h}{\text{RMSE}(\mathbf{x}, \mathbf{y})} \right) \geq 0$$

Nevertheless, with RMSE and PSNR, a small constant bias or a variation of contrast between two otherwise identical images can produce a significant error measure, despite the two images being perceptually very close. For this reason, we complete these measures with UQI and SSIM.

UQI

The UQI index of [Zhou Wang and Bovik \[2002\]](#) measures the similarity between two images under a perception-based model:

$$\text{UQI}(\mathbf{x}, \mathbf{y}) = \frac{4\sigma_{xy}\bar{x}\bar{y}}{(\sigma_x^2 + \sigma_y^2)(\bar{x}^2 + \bar{y}^2)} \in [0, 1],$$

where \bar{x}, \bar{y} are respectively the mean value of \mathbf{x} and \mathbf{y} , and σ_{xy} , σ_x^2 and σ_y^2 are respectively the variances of \mathbf{x} , \mathbf{y} , and their covariance.

SSIM

The SSIM shares the same idea of introducing perception-based models to compare two images, but integrates the variances and covariances locally over a sliding window $\Omega(p)$. The SSIM writes:

$$\text{SSIM}(\mathbf{x}, \mathbf{y}) = \frac{1}{\#\mathcal{P}} \sum_{p \in \mathcal{P}} \frac{(2\bar{x}_{\Omega(p)}\bar{y}_{\Omega(p)} + C_1)(2\sigma_{xy_{\Omega(p)}} + C_2)}{(\bar{x}_{\Omega(p)}^2 + \bar{y}_{\Omega(p)}^2 + C_1)(\sigma_{x_{\Omega(p)}}^2 + \sigma_{y_{\Omega(p)}}^2 + C_2)} \in [0, 1],$$

where \mathcal{P} is the set of the centers p of the used windows $\Omega(p)$ of size 7×7 , $\bar{x}_{\Omega(p)}$, $\bar{y}_{\Omega(p)}$ are the means over $\Omega(p)$ of \mathbf{x} and \mathbf{y} values respectively, and $\sigma_{x_{\Omega(p)}}$, $\sigma_{y_{\Omega(p)}}$, and $\sigma_{xy_{\Omega(p)}}$ are the variances and covariance. Finally, the constants C_1 and C_2 are computed from Δ_h as $C_1 = (0.01\Delta_h)^2$ and $C_2 = (0.03\Delta_h)^2$ for numerical stability.

This is the version of SSIM specified in [Wang and Sheikh \[2004\]](#) with (according to their notations) $\alpha = \beta = \gamma = 1$. By computing the variances, covariance and mean values on a set of windows covering the whole image, SSIM incorporates comparison measurements of luminance, contrast and structure of two images, and aims at taking into account important perceptual phenomena in its evaluation.

We start the next section by using these metrics to have an overview of the best performances of our approach.

6.4 Experimental results

In this section we present the interest of our approach by comparing the depth map obtained with different parameters and different neighborhoods, both with qualitative and quantitative evaluation methods. In the qualitative approach, we compare visually the depth maps and neighborhood quality metrics for different values of the regularization coefficient α and different types of neighborhoods, for both RORPO and TV. In the quantitative evaluation, we use the evaluation metrics proposed in Section 6.3.3 to discriminate the best results for each parameter and compare them for all the images of our dataset.

6.4.1 Global segmentation results

We now test our approach extensively on a selection of images from the Middlebury college dataset. For comparing the performances of each neighborhood construction on each scene, we compute the PSNR, the UQI and the SSIM, for each depth map with different sets of parameters. Even if we compute each neighborhood for a set of values of the regularization coefficient α , we represent the performance of each neighborhood with its best result for each measure. We do this both for computing the depth at superpixel level, and at pixel level, in Figure 6.10.

This allows for a quick numerical comparison of the performances over different scenes. Overall, the perfect neighborhood seems to perform well against most of the neighborhood implemented in this experience, both at pixel and superpixel level, even if some exceptions exist. In general, Stawiasky's neighborhood performs better at pixel level than at superpixel level.

In a complementary fashion, our anisotropic approach generally seems to bring less satisfactory results at pixel level, and even tends to fail drastically on some isolated cases, for instance in *Wood21*, *Bowling* or *Plastic*. In other cases, these anisotropic neighborhoods at pixel level are quite close in term of performances to perfect and Stawiaski and Decencière [2011], sometimes even seems to outperform one or two of them, for some of the metrics, but the results are somewhat mitigated.

At superpixel level however, anisotropic and isotropic performances for depth reconstruction seem to be more balanced, to the benefit of cardinal-based neighborhood. For exhibiting the benefits of our approach, we turn our attention to the cases where the performances highly depend on the type of neighborhood. For instance, *Flowerpot*, *Reindeer* and *Lampshade* present high variations in the performances, to the advantage of cardinal-based neighborhood, both with TV and RORPO. We also draw your attention to the more difficult cases, that seem to be the same scenes than at pixel level.

For further scene analysis, we present the corresponding error maps and neighborhood quality maps.

In general however, shape-based neighborhood seems to often yield average or poor results, both at pixel and superpixel level, both with RORPO and TV. For this reason and because the number of images we can display in a figure is limited, we decide to let aside shape-based neighborhood and to compare only cardinal-based and target-based neighborhoods against Stawiaski and Decencière [2011] and the perfect one in what follows.

Qualitative analysis

Successful scenes

We focus on the scenes *Reindeer* and *Lampshade* for analysis and validation of the interest of our approach. Through the display of error maps and neighborhood quality maps, for each neighborhood with multiple values of α , we discuss in this section what makes the strengths of each approach in these two examples.

Reindeer The ground truth of this scene, called *Reindeer1* in the Middlebury College dataset, is presented in Figure 6.11.

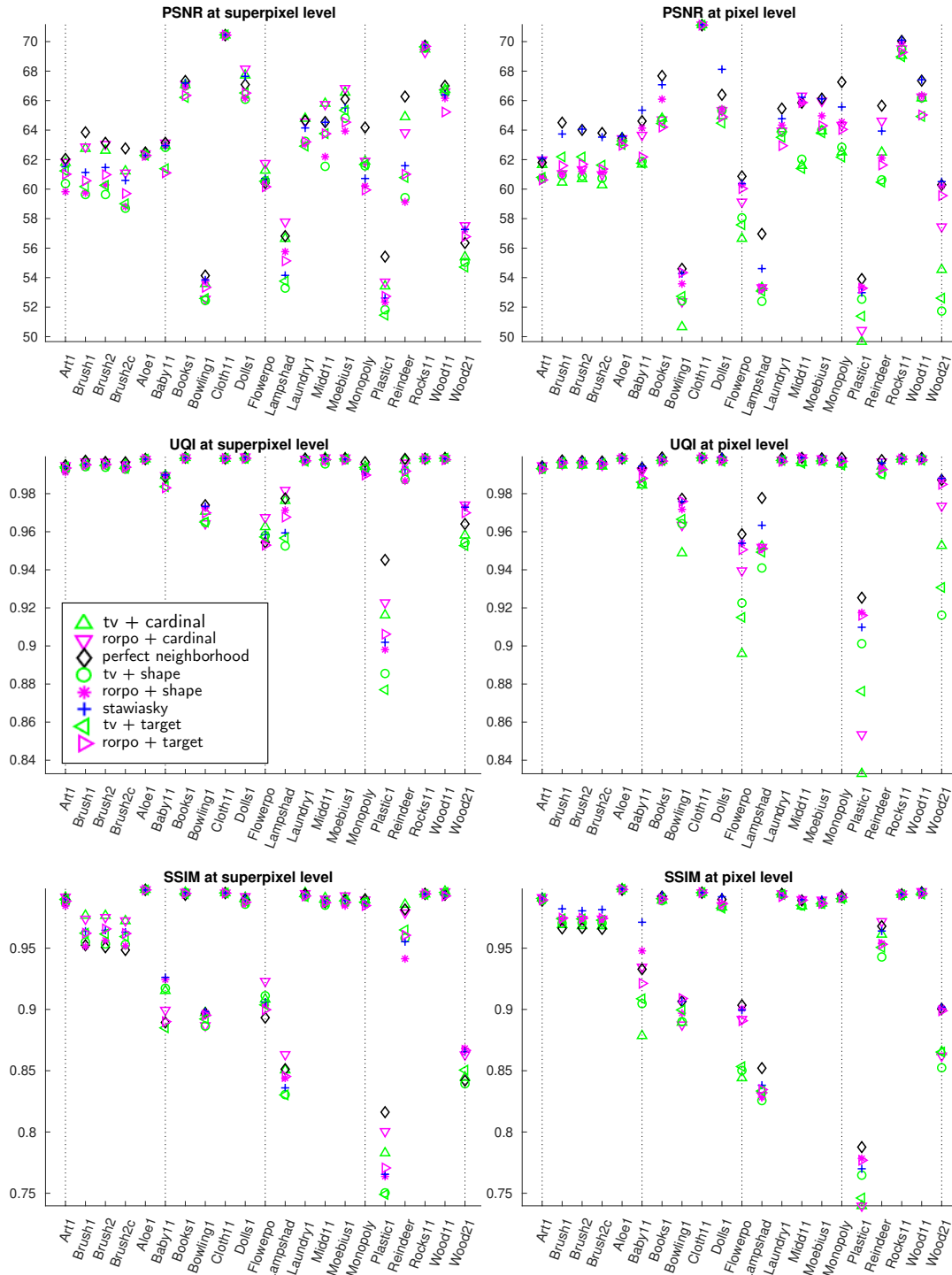


Figure 6.10 – Per scene best results at superpixel (left column) and pixel level (right column). Each row plots the best results for each neighborhood construction and for each scene, among all experiences, respectively to a quality measure: From top to bottom, PSNR, UQI and SSIM. For all of these measures, the higher is the value, the better is the result. Values obtained with TV are colored in green, and with RORPO in purple, while our perfect neighborhood and Stawiasky’s isotropic neighborhood are represented respectively with black diamonds and blue crosses marks. Cardinal-based, target-based and shape-based neighborhoods’ performances are represented with four types of triangles, circles and asterisks (see more details in the legend). Since shape-based neighborhood results generally seems to be slightly below the other approaches, we removed it from the qualitative comparison in the next figures for better readability.



Figure 6.11 – The ground truth and the all-in-focus image for the scene *Reindeer* of the dataset.

In this image, we have, from the bottom to the top, a moving box with a thin textured rope in the foreground, topped by a reindeer that has antlers above his head and holds gifts. The reindeer lays below the background, and is therefore very contrasted in the depth map, whereas the rope is a challenging thin structure that lies just a few centimeters below the moving box. A second moving box lies on the right hand side of the image, in the foreground. Both moving boxes exhibit a relatively poorly textured surface, for which obtaining the blind estimate of the depth may be challenging.

Both in the middle-ground and in the middle of the image, a carved head seems to rest on a sofa, haloed by the handle of a wicker basket. On the top of the image, the background is composed of a tote bag and the back of the sofa.

The construction of superpixels with ETPS yields the partition of the image visualized in Figure 6.12. We note that the choice of the superpixel algorithm used may have a high impact on the decomposition of the image in sites and therefore on the construction of the guidance map, the neighborhoods, and the resulting segmentation. We formulated in the beginning of our works some qualitative requirements for the superpixels to be used. Among them, there are for instance the *regularity* of the superpixel lattice, especially in homogeneous areas, the *adherence* to boundaries, and the *regularity* of the size of the superpixels.

While ETPS seems to fulfill these prerequisites and allows the construction of suitable neighborhoods for our application, more advanced experiments could be carried out on this topic and are left for future works.

We present in Figure 6.13 the depth maps obtained by varying of the regularization coefficient for each of the neighborhoods considered, namely: The *Perfect* neighborhood, the *Cardinal-based* neighborhood, the *Target-based* neighborhood, and *Stawiasky's* neighborhood. For the anisotropic neighborhoods, we compute two versions, based on TV and RORPO.

For this scene, we note in the blind segmentation that the moving boxes are very noisy. This noisy aspect is an expected consequence of the noise added in the blind depth. This occurs in low-texture areas, designed for avoiding any constant bias that could be introduced by the sharpness operator when its response is too small. There are also errors of depth on the reindeer, on his head and on the chest.

On the contrary, the rope, the wicker handle and the head, look correctly estimated. The carved head also looks quite satisfying, as well as the top-left part of the background.

In the end, the challenges of the regularization is to preserve these correct depths while smoothing the erroneous areas.

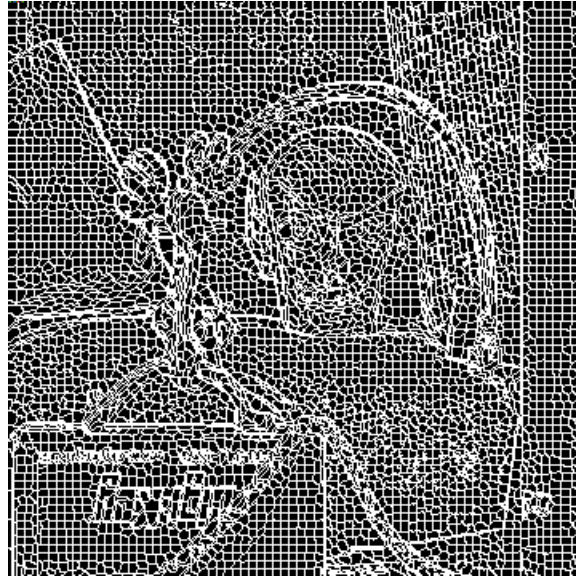


Figure 6.12 – The superpixels generated by the ETPS algorithm (Yao et al. [2015]) for the scene *Reindeer* of the dataset. These superpixel exhibit relatively regular shapes in areas that are homogeneous, and follow the edges of the objects of the scene, which makes them a good candidate for our experiments.

When looking at the board, from blind segmentation to more regularized depth maps until the best result of each column, we see that the first parts to be regularized are the noisy areas in the foreground on the moving boxes, thus improving the quality of the depth reconstruction according to our quality criteria. For target-based and Stawiaski and Decenci re [2011]’s neighborhoods however, the antlers start to deteriorate (for $\alpha = 0.5$), while noisy areas of the boxes in the foreground are not correctly segmented.

These errors are easier to localize in Figure 6.14. The best result is achieved for $\alpha = 2$ with RORPO and cardinal-based neighborhood. When increasing the regularization parameter further, we can see here that the cardinal-based neighborhood and the perfect neighborhood still quite preserve the antlers. This suggests that these neighborhood exhibit better performances in these areas.

As expected when looking at the boxplot Figure 6.15, the distribution of the errors in the perfect neighborhood is very small and its mean is a null error, which is encouraging. The other neighborhoods share these characteristics, and the distinctions between them are mostly a question of positioning of the outliers. For instance, target-based neighborhood with TV-based guidance map has a large amount of consequent errors, localized in the same area as the other target-based neighborhood, that seems to be related to errors in red/orange on Figure 6.14 on the antlers. The boxplot also signals more outliers on RORPO-cardinal-based neighborhood than on TV-cardinal-based neighborhood, which favors the latter one, as confirmed both by PSNR, UQI and SSIM.

To go further, we finally compare the neighborhood themselves through the indicators we have implemented in Figure 6.16. First, we can compare the orientations. On the TV-based guidance map, we can see clearly piecewise smoothly varying orientations, with a few set of outliers. Due to our implementation, we recall that the TV is only performed over a surface computed from the maximum of sharpness, and that the smoothness of orientations is therefore only ensured over the smooth parts of that surface. A side effect of the voting algorithm is that sites close to the border of the image tend to have an orientation that is parallel to the edge of the image. On a larger image or with the appropriate border extrapolation, such effect would not impact the orientation estimation; as a result we ignore this effect in our analysis. However, looking at its effect near the angles of the image still gives us hints about the behavior of TV near edges of compact structures: In a counter-intuitive manner, the estimated orientation in the top left corner of the image is the orientation of the diagonal of the image, from the bottom right to the top left. This indicates that the estimated orientations near the edges of high curvature of a compact structure will tend to

be orthogonal to these edges. This problem of orientation near edges with TV is visible near the basket handle and the right side of the rope in the comparison of cardinal-based neighborhoods in the second and third row of Figure 6.16.

With RORPO, the approach is different, and it visually produces saliency and orientation maps that highlight better the sharp edges of the objects. This is consistent with the fact that with RORPO, we seek for high values of the sharpness operator (the SMLAP) for performing the path opening and discriminating thin structures. A drawback is that the map of orientations seems to lack a bit of smoothness.

If we look at the rope in the foreground, we see without any doubt that both the method converge towards a set of orientations that seems to fit the curvature of the rope: The left side is colored in purple and red corresponding to an angle of about $\pi/2$ with the horizontal axis \mathbf{e}_1 . The right side is colored in shades of green to blue that corresponds to an angle of about $-\pi/2$ with \mathbf{e}_1 .

The same idea goes with the edges of the boxes, the four feet of the deer, its neck, and the wicker handle, that seem to have a correct orientation estimation, even if it is less clear with the RORPO.

However, when looking at the direct surroundings of the handle in the TV guidance map depth map, we can notice the orthogonal orientations on the edges of compact structures that we mentioned. This effect is especially visible above the handle, and is also noticeable above the left foot of the reindeer in the surroundings of the right part of the rope.

We also note that the sharp details at equal depths in the image are not impacting the TV guidance map, while they appear in the RORPO guidance map as for example in the edge of the tote bag.

Finally, the second and third row of Figure 6.16 confirm the location of the weaknesses of the anisotropic approaches: We can see the antlers of the reindeer are the main area where, for all the approaches, the neighborhood construction seems delicate, and especially for target-based and Stawiaski and Decenci re [2011]’s neighborhood. This both corroborates the observations on the error maps and validates the interest of our neighborhood quality criteria.



Figure 6.13 – Depth maps obtained for the scene *Reindeer1*, for different neighborhood strategies and different values of α , which are indicated on the left hand side. The arrows indicate the direction of increasing α , from top to bottom. The neighborhood strategy used is indicated above of each column, and the best depth map obtained (with respect to the PSNR) for each neighborhood is indicated with a star on the upper right corner of the depth map. According to the PSNR, the neighborhood that gives the best result here is the TV cardinal-based neighborhood, after the *perfect* one. The differences are mainly visible on the antlers of the reindeer that are well preserved by cardinal-based neighborhoods. For interpreting the results, we also print the error map on Figure 6.14.

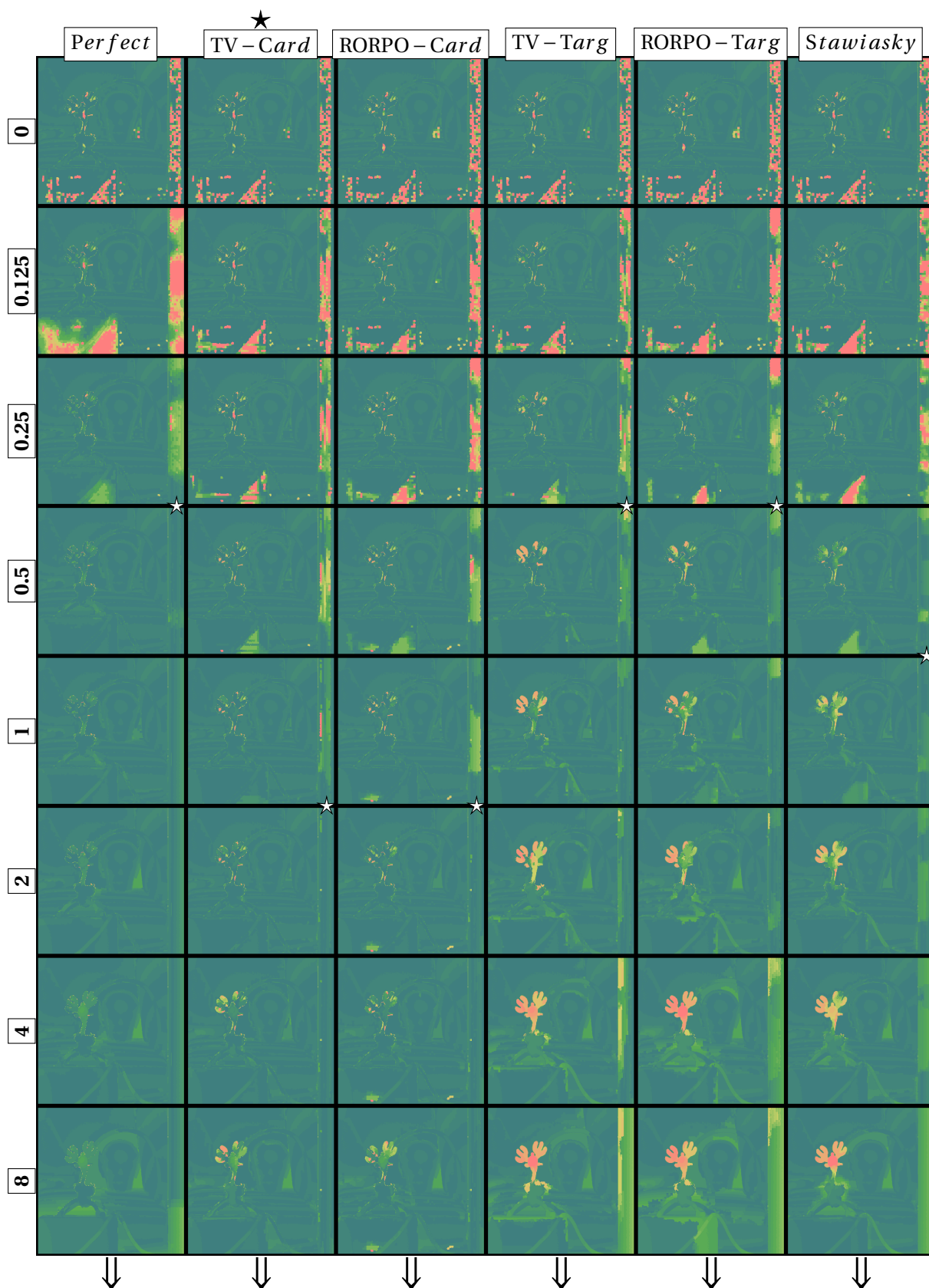


Figure 6.14 – Error of depth maps obtained for the scene *Reindeer*, for different neighborhood strategies and different values of α , which are indicated at the top and on the left hand side, respectively. The arrows indicate the direction of increasing α , from the top to bottom. We can see that the best results are obtained by cardinal-based neighborhoods, since the depth of the moving box at the right hand side seems badly estimated for the other ones while the antlers of the reindeer get quickly degraded.

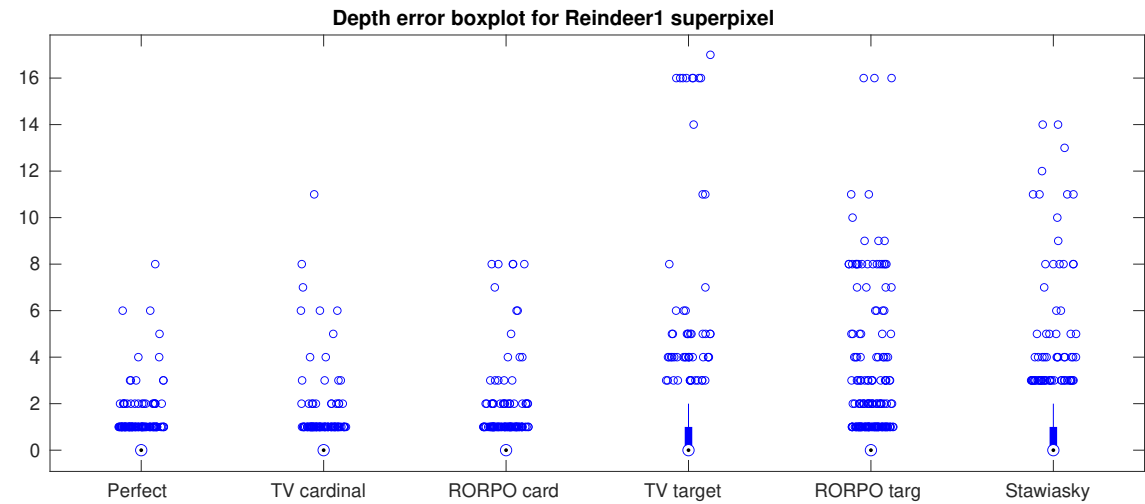


Figure 6.15 – Reindeer error map boxplot corresponding to the best results of Figure 6.14. According to these boxplots, the distribution of errors tends to present the *Perfect* and cardinal-based neighborhoods as the best candidates for this scene.

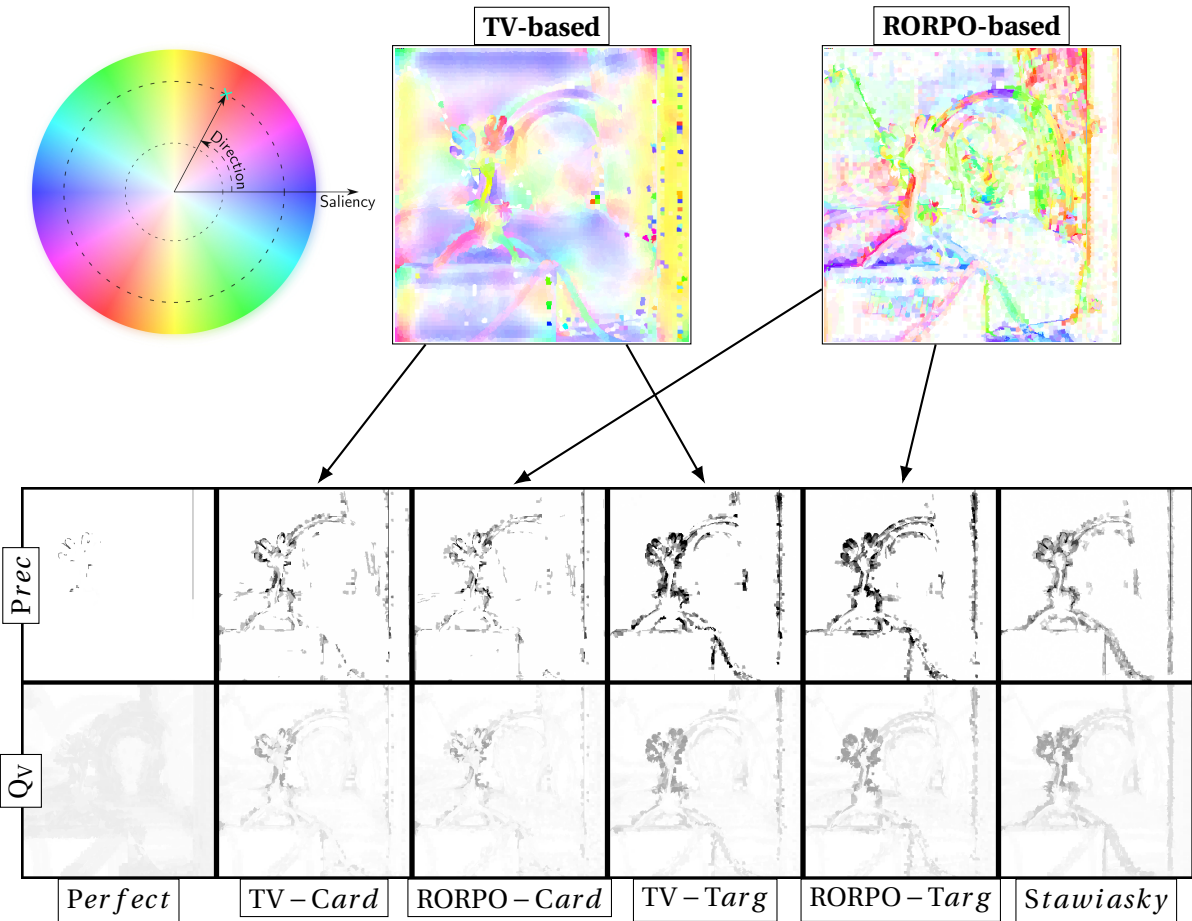


Figure 6.16 – The guidance map from RORPO and TV are displayed on the first row and exhibit the behavior of these two algorithms for estimating the saliency and orientation of structures of the image. While TV looks smoother in terms of orientation, RORPO highlights better the sharp areas of the scene. The second and third rows show comparative quality of the neighborhood construction with respect to $Prec$ and Q_v respectively. For instance note that the area near the antlers effectively indicate a significant difference of quality between the neighbors.



Figure 6.17 – The ground truth and all-in-focus image for the scene *Lampshade* of the dataset.

Lampshade This scene features a set of boxes in the background, with a plastic folder and a piece of wood. On the left hand side, the foot of the lamp is a thin structure that contrasts with the wall on the background, and in the foreground in the middle of the image lies a lampshade. By nature, this scene does not feature much thin and complex structures, and thus, the need for anisotropic neighborhood is not clear, except for the preservation of the foot of the lamp.

However, when looking at Figure 6.10, anisotropic neighborhoods seem to perform well in that situation, and bring an important added value against the isotropic neighborhood. We can check what happens visually in Figure 6.18. First of all, in the blind segmentation, we see two noisy areas, the lampshade excepted in its edges and the whole plastic folder in the background. Thus, the challenge of regularization is twofold: Firstly, the goal is to strongly regularize those errors in the depth estimate without losing the lamp foot, and secondly, the regularized mean value of the depth in the lampshade section must approach the actual depth of the lampshade.

The best results, also indicated by a star in Figure 6.19, are obtained with cardinal-based neighborhoods (and, indeed, perfect neighborhood). Isotropic neighborhood fails at both the challenges, while target-based approaches seem to at least preserve a bit the lamp foot while the parameter α increases. Shape-based and perfect neighborhoods succeed in preserving the thin structure while at the same time allowing for a good regularization of the lampshade, even if most the depth error concentrates on this part of the scene.

Boxplot pictured in Figure 6.20 confirm that the depth errors in this image tends to be larger, since even for perfect neighborhood, depth error as big as nine frames in the set of defocused images still belong to the ninth decile of the distribution. The mean error for [Stawiaski and Decenci re \[2011\]](#)’s neighborhood even reaches one frame.

Figure 6.21 allows to compare the neighborhood quality for each strategy. We can clearly see on the colored map of orientations the side effect of TV near the borders of the lamp foot, and its consequences on both of our neighborhood quality criteria. Here, RORPO performs a bit better than TV with respect to these criteria, and cardinal-based neighborhood performs better than target-based and [Stawiaski and Decenci re \[2011\]](#)’s neighborhood. Once again, it is totally reflecting the results shown by the depth error maps and the Table 6.4 that show the best result achieved with RORPO cardinal-based neighborhood.

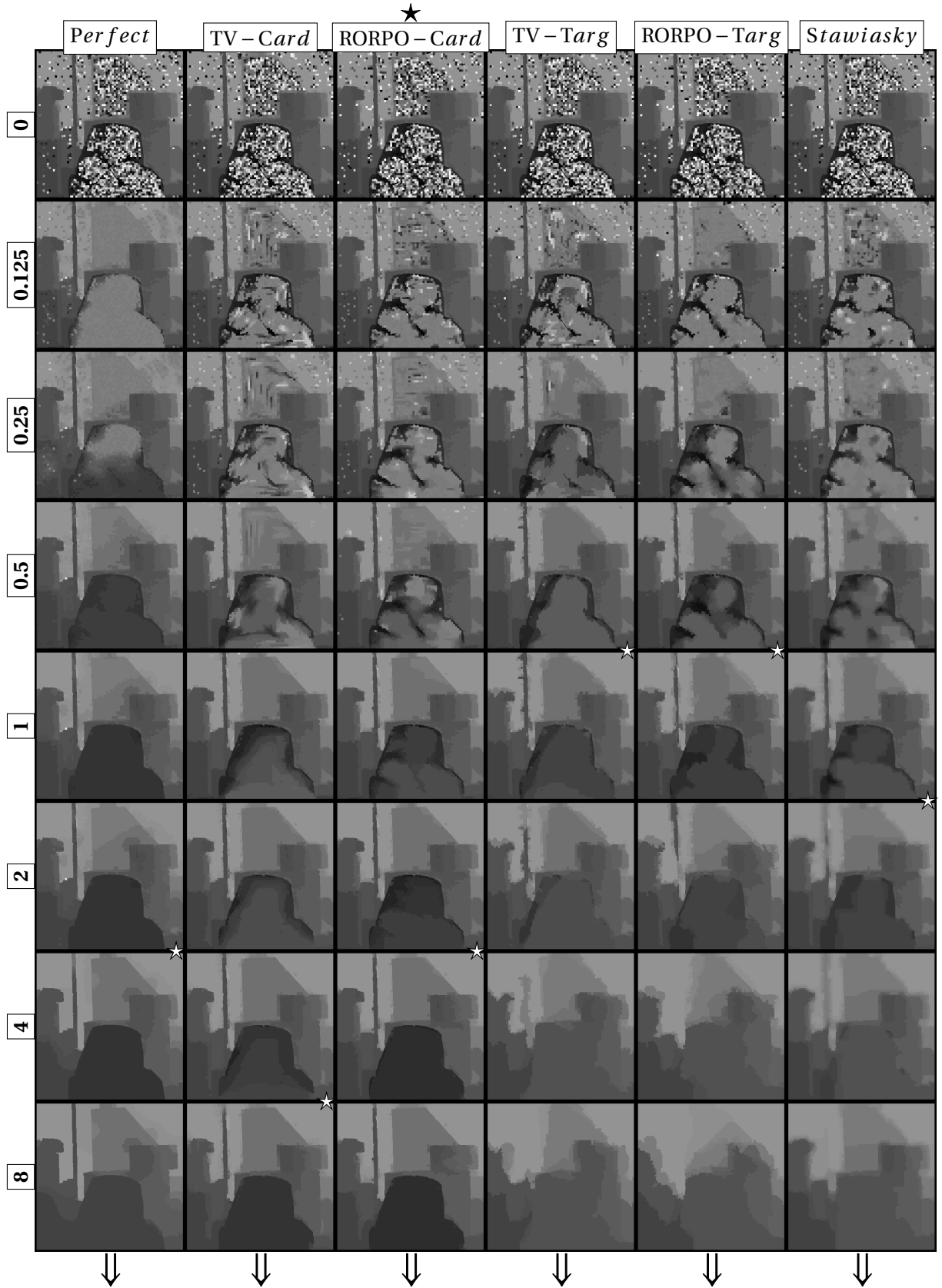


Figure 6.18 – Depth maps obtained with scene *Lampshade*, for the neighborhood strategies indicated above and multiple values of α indicated at on the left hand side. For each neighborhood strategy, a star on the top right corner of a depth map indicates the best result according to PSNR. This scene is challenging since the initial depth is very noisy because of homogeneous regions. The advantage of anisotropic neighborhood here is to preserve the sharp edges of the scene while α increases.

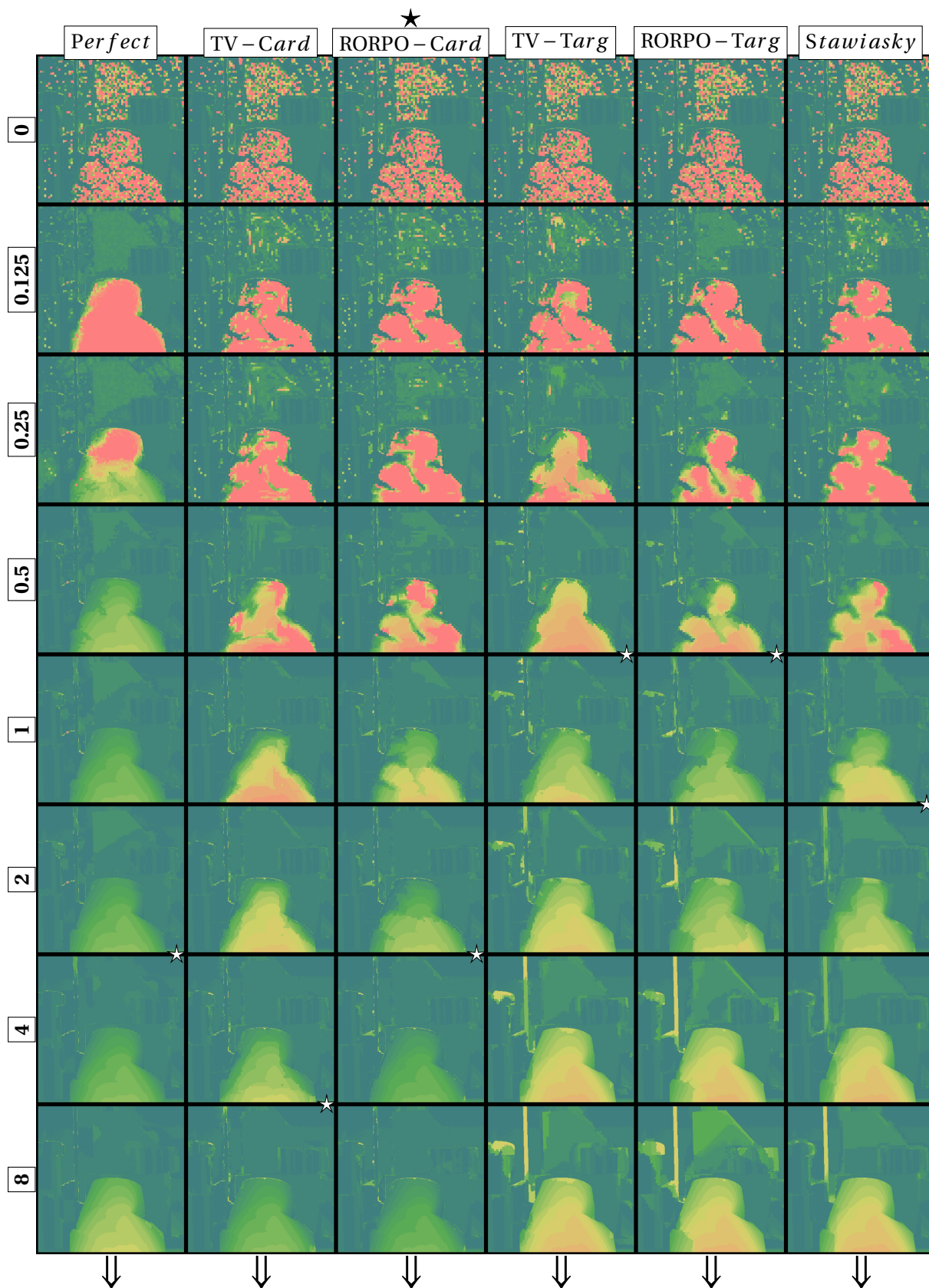


Figure 6.19 – Depth error maps obtained with the scene *Lampshade*, for the neighborhood strategies indicated above and multiple values of α indicated at on the left hand side. For each neighborhood strategy, a star on the top right corner of a depth map indicates the best result according to PSNR. One challenge of this image, additionally to the preservation of sharp edges and thin structures, is the convergence of the regularized depth map to the right depth, without bias induced by the noise. Such challenge may however fall to the model formulated in Equation (2.3).

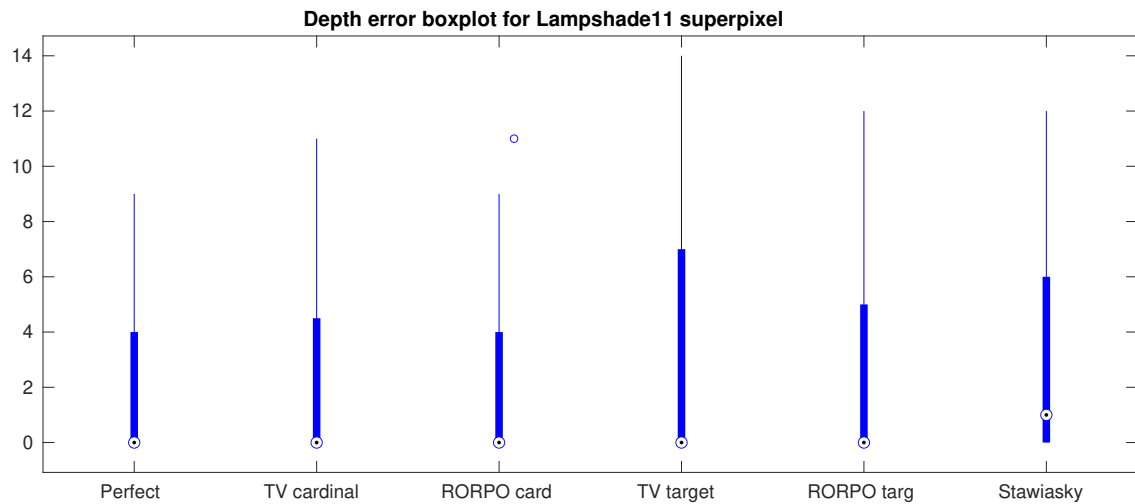


Figure 6.20 – Lampshade error map boxplot corresponding to the best results of Figure 6.19. Compared to previous experiments, the interquartile range is much higher as well as the ninth decile value. This explains the small number of outliers in this case. The results are somewhat similar but RORPO-based and perfect neighborhoods seems to perform better, followed by TV-based and Stawiasky’s neighborhood.

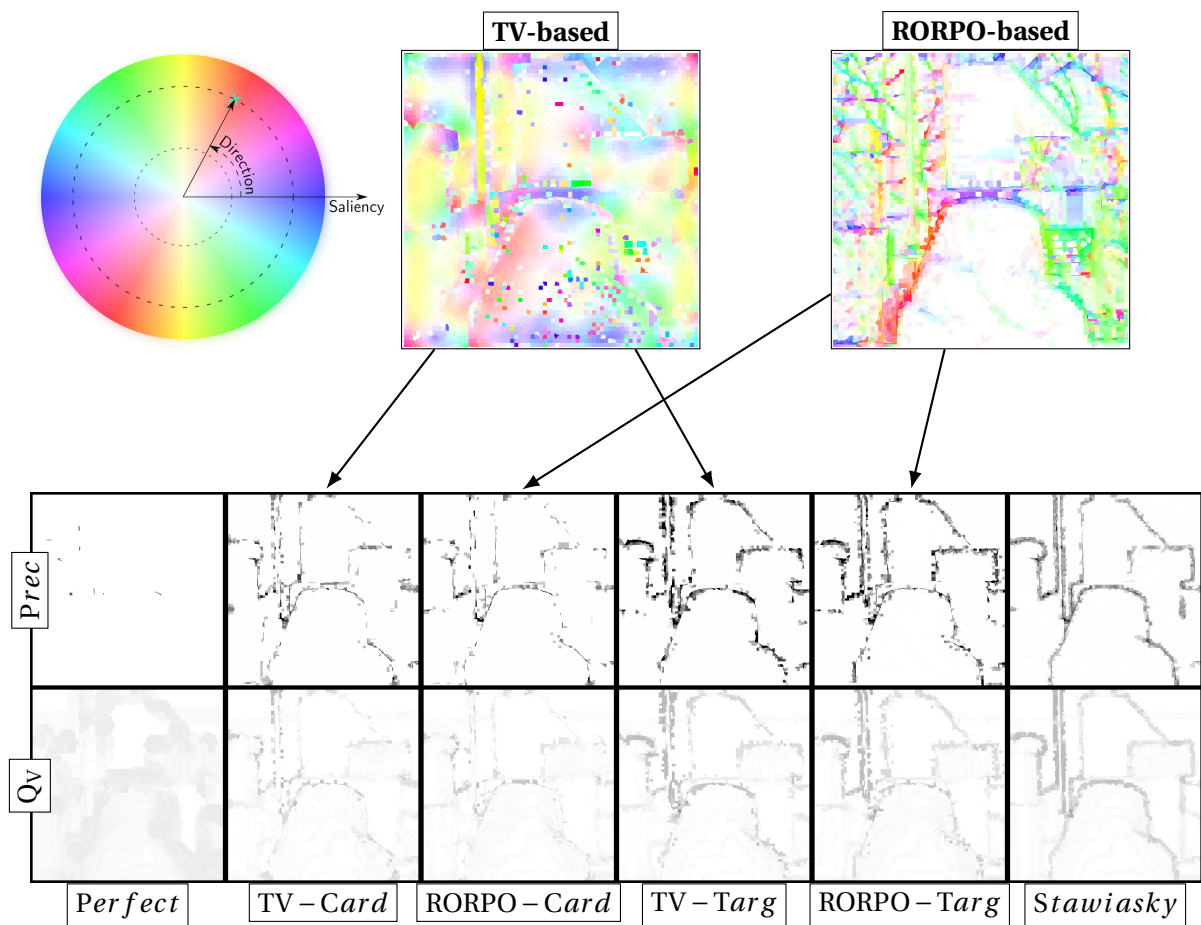


Figure 6.21 – The guidance map computed for the Lampshade scene from TV and RORPO are presented in the first row. The qualitative criteria for the each neighborhood are displayed on the second and the third row. We note that the cardinal-based neighborhood still present better results, especially when looking at the foot of the lamp on the left side of the scene.

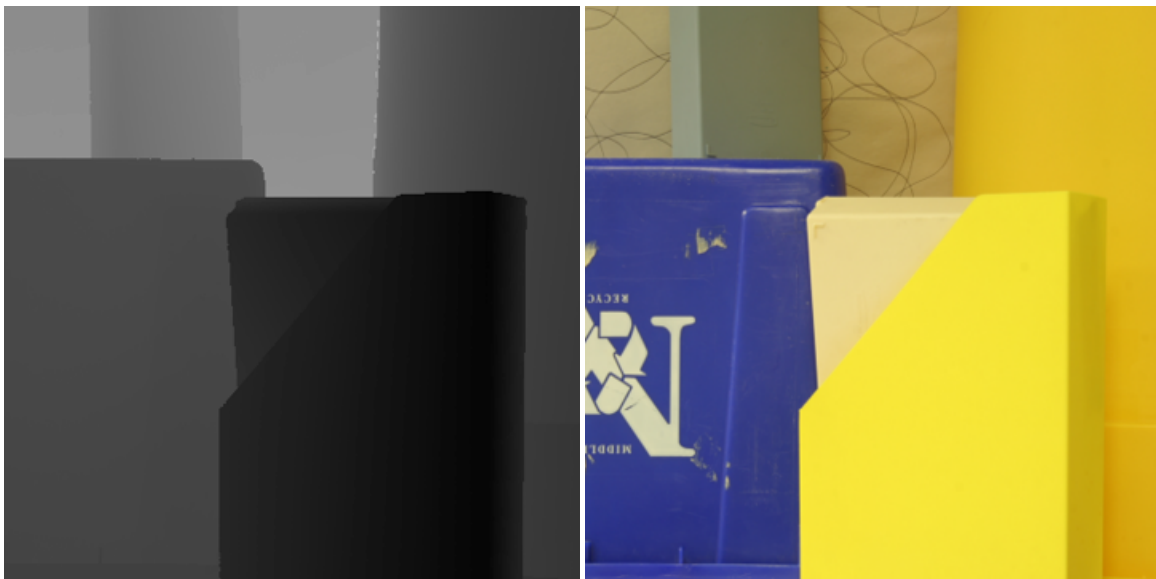


Figure 6.22 – The ground truth and all-in-focus image for the scene *Plastic1* of the dataset.

Challenging scenes

Now we present more challenging scenes, for which the RMSE criterion revealed a general failure in regularization or gave a clear advantage to the isotropic neighborhood against ours. The scenes *Plastic1* and *Bowling11* are one of those. We also picked one image to present the approach at pixel level, *Art1*, since it presents multiple thin structures and an interesting variety of shapes.

Plastic The Plastic scene is pretty simple in appearance: A yellow plastic folder lies in a foreground, below a blue recycle bin, a light blue wooden board and an unidentified yellow in the middle-ground. The whole is leaned against textured wall that constitute the background of the image.

The main issue is that the image is actually poorly textured, with a lot of simple geometric structures, mostly quadrilaterals with straight edges. In this situation, the benefits of anisotropic neighborhoods could be limited. However, according to Figure 6.10 and Table 6.5, RORPO-cardinal-based neighborhood still slightly outperforms the other approaches at superpixel level. At pixel level, the results are even more mitigated.

Here in Figure 6.23, we present the results at superpixel level. As expected with homogeneous regions, the blind segmentation here has lots of errors and the challenges are quite the same as in the scene *Lampshade*, but with a higher level of noise. As a result, the plastic folder's depth tend to converge to the recycle bin's depth, or even with the yellow object on the right hand side a bit further. For that reason, there is a bias in the detection and relying only on the numerical performances may indicate as a best representative some experience that do not look successful. For instance, the isotropic best result here seems to be the over regularized result with $\alpha = 32$, even if the shapes of all the individual objects of the scene have disappeared.

Similar phenomenon happens for all the types of neighborhoods in that scene: Blurring the plastic folder always increases the accuracy of its depth positioning, while our eyes judge differently. Indeed, we have the tendency to mentally correct the bias in brightness and to rely on the edges of the objects to distinguish them. Figure 6.24 allows to disambiguate a bit by bringing color into the error map, but in general, this example seems too hard to solve simply with regularization, since even the perfect neighborhood fails at it.

The model could however be modified to improve a bit the behavior of the algorithm. While the low sharpness randomized sites are already penalized in their data term by the weighting function W_s in Equation 6.2, an improvement could be to manually set it to a very low value or to set $W_s = 0$ when, for any $s \in \mathcal{S}$: $\max_{t \in \mathcal{H}_s} (\mathbf{f}(t)) < \kappa_{\mathbf{f}}$. However, this corresponds to removing a constraint

on the optimization, and one should care about ensuring that by transitivity, each site should have in its neighbors a site that connects to a site for which $W_s \neq 0$. Otherwise, the problem would become ill-posed again. Ideally, this could be done by computing anisotropic neighbors in the surrounding of the homogeneous regions, and isotropic neighbors inside them, at the risk of lowering the recognition of poorly textured thin structures.

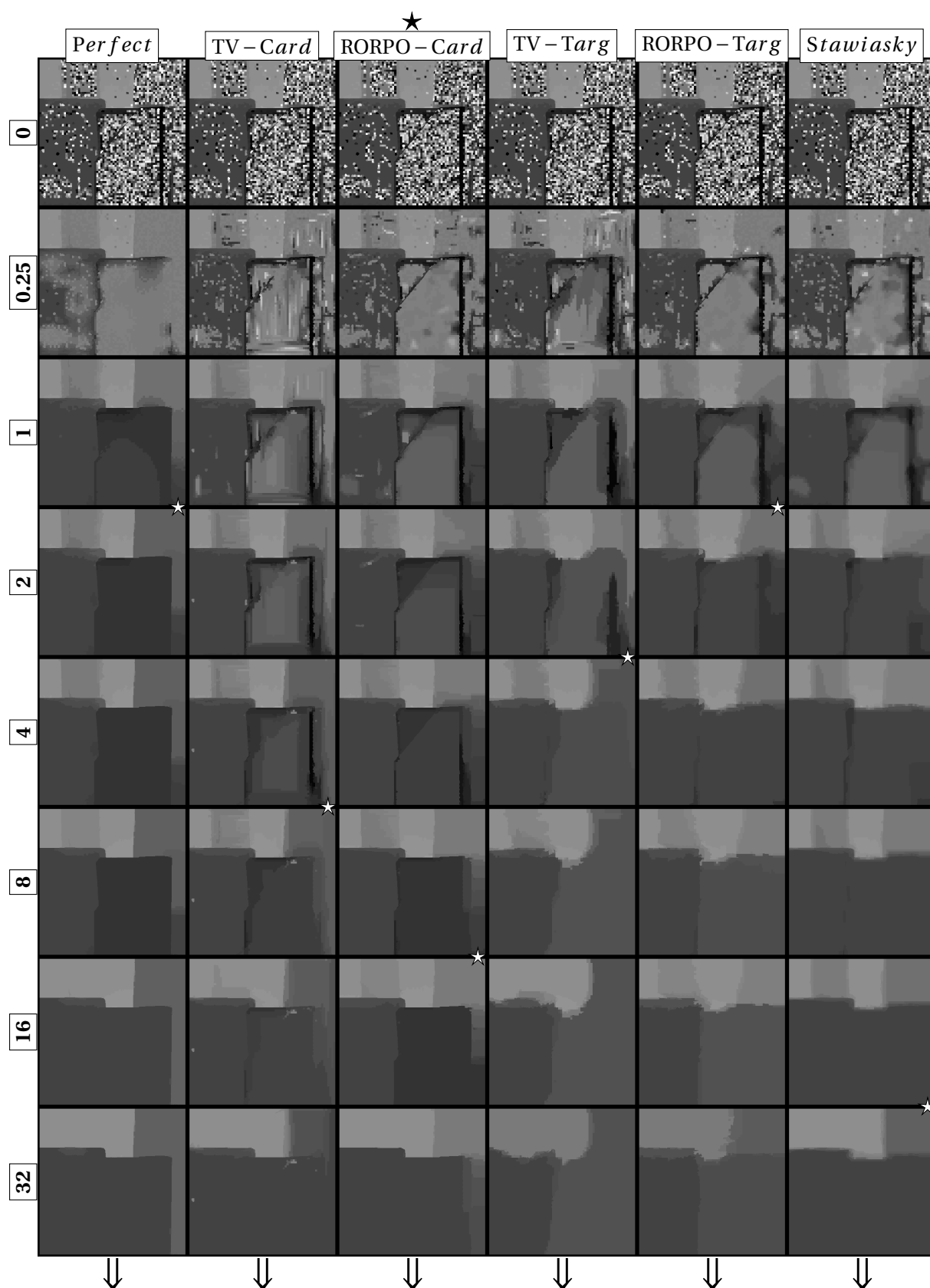


Figure 6.23 – Depth maps obtained with scene *Plastic*, for the neighborhood strategies indicated above and multiple values of α indicated at on the left hand side. For each neighborhood, a star on the top right corner of a depth map indicates the best result according to PSNR. In this scene, some of the best results are achieved while seemingly loosing some information on the image. This is mostly because since the initial depth map is very noisy, the depth inside the plastic folder in the middle is badly estimated and over-regularizing it yields best results.

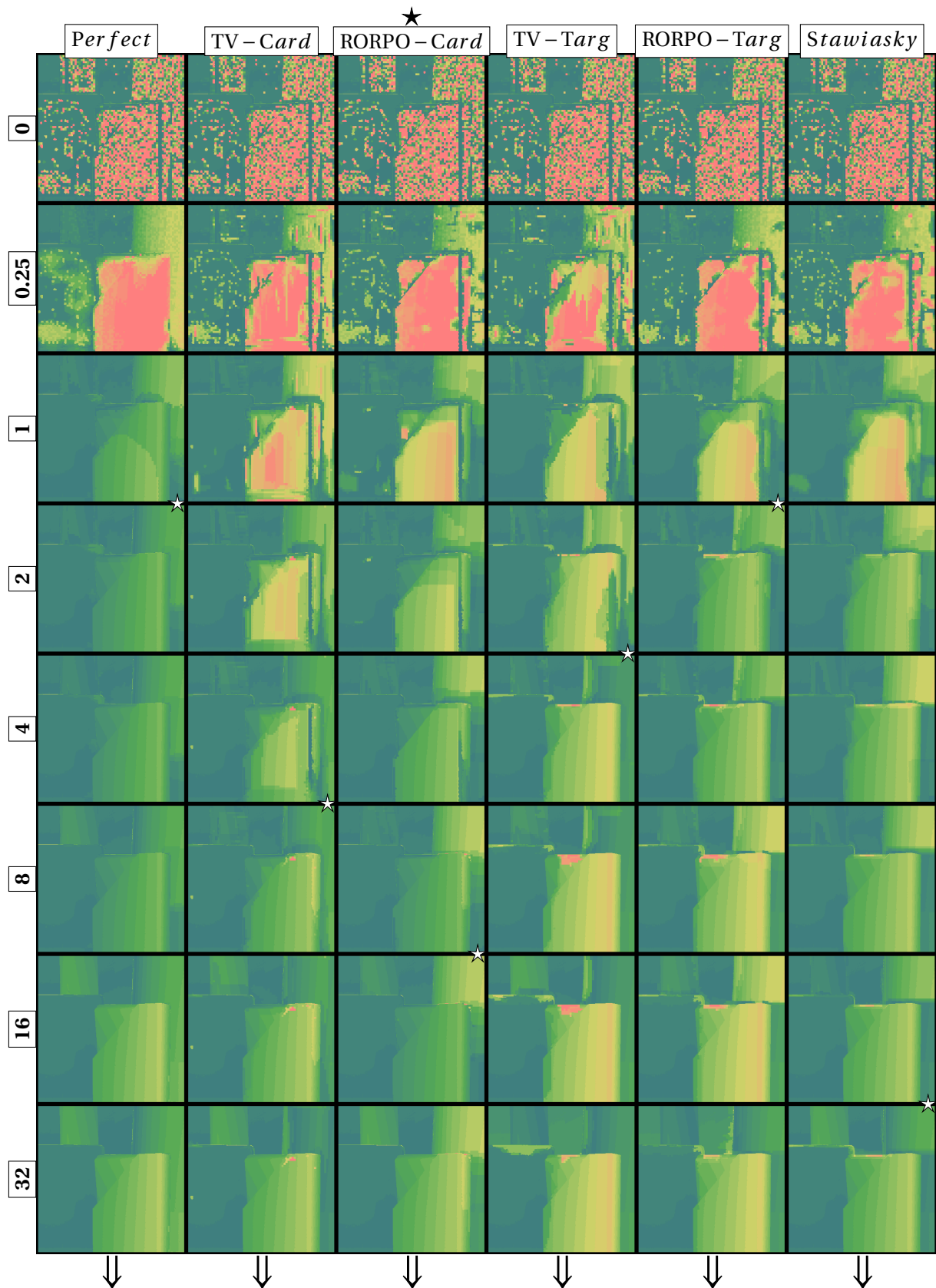


Figure 6.24 – Depth error maps obtained with scene *Plastic*, for the neighborhood strategies indicated above and multiple values of α indicated at on the left hand side. For each neighborhood, a star on the top right corner of a depth map indicates the best result according to PSNR. These maps confirm the issue with depth initialization, which is a problem that regularization alone is not likely to solve easily.

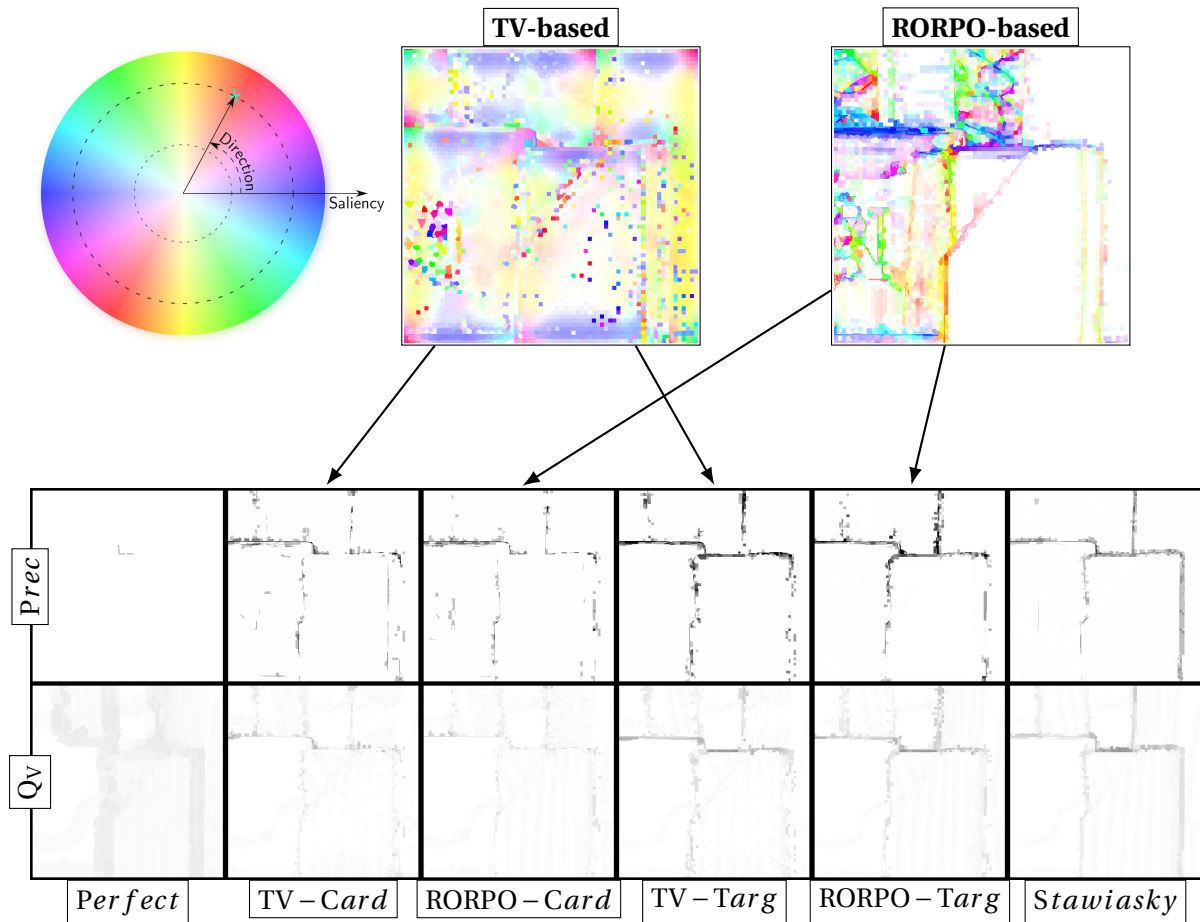


Figure 6.25 – The guidance map computed for the Plastic scene from TV and RORPO are presented in the first row. The qualitative criteria for the each neighborhood are displayed on the second and the third row. While the orientations are quite well estimated, this scene features regular shapes that do not seem challenging to segment with any of the neighborhoods, and the performances appear relatively similar for anisotropic and isotropic neighborhoods.

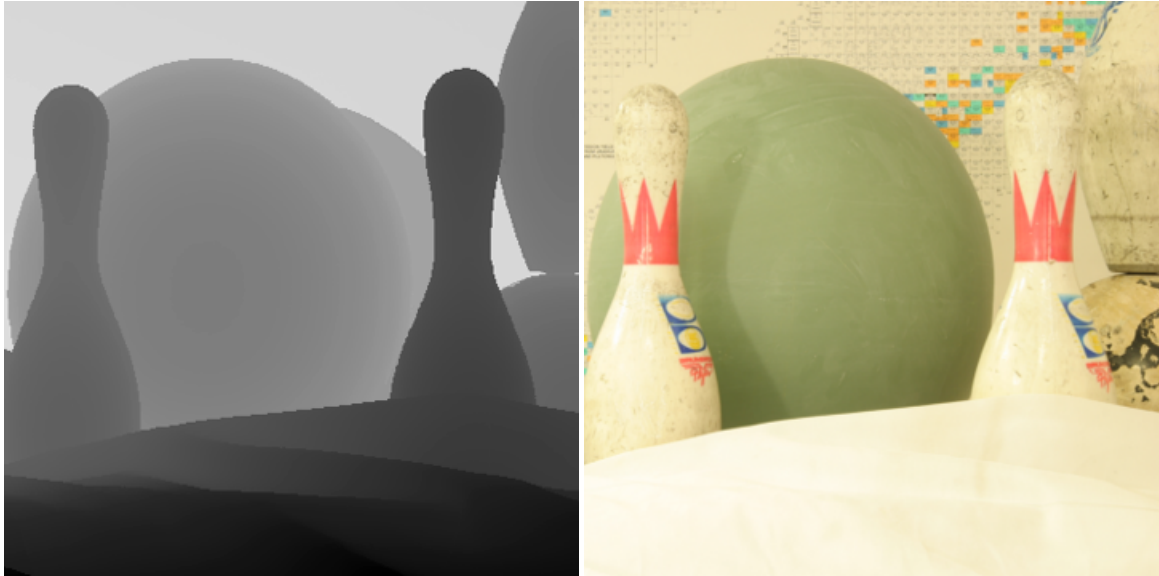


Figure 6.26 – The ground truth and all-in-focus image for the scene *Bowling* of the dataset.

Bowling The *Bowling* scene shown in Figure 6.26 features a green bowling ball and three pins below a partly textured background. In the foreground, there is a white piece of fabric that is poorly textured. As confirmed by blind segmentation in Figure 6.27, the challenging regions are mainly the piece and fabric in the foreground, the bowling ball in the middle ground and the homogeneous part of the background.

When looking at Figure 6.28, most the errors in this scene are due to a bias in the estimation of the piece of fabric’s depth. The best results are obtained with a strong regularization of this area, therefore leading to a quick deterioration of the edges of the objects, especially with isotropic and target-based neighborhoods, giving advantage to the RORPO cardinal-based neighborhood. This advantage is confirmed in Figure 6.29, where the neighborhoods seem overall quite satisfying, despite the bad quality of the depth reconstruction’s results.

In a way quite similar to the scene *Plastic*, we can suggest adapting the model to help with depth regularization.

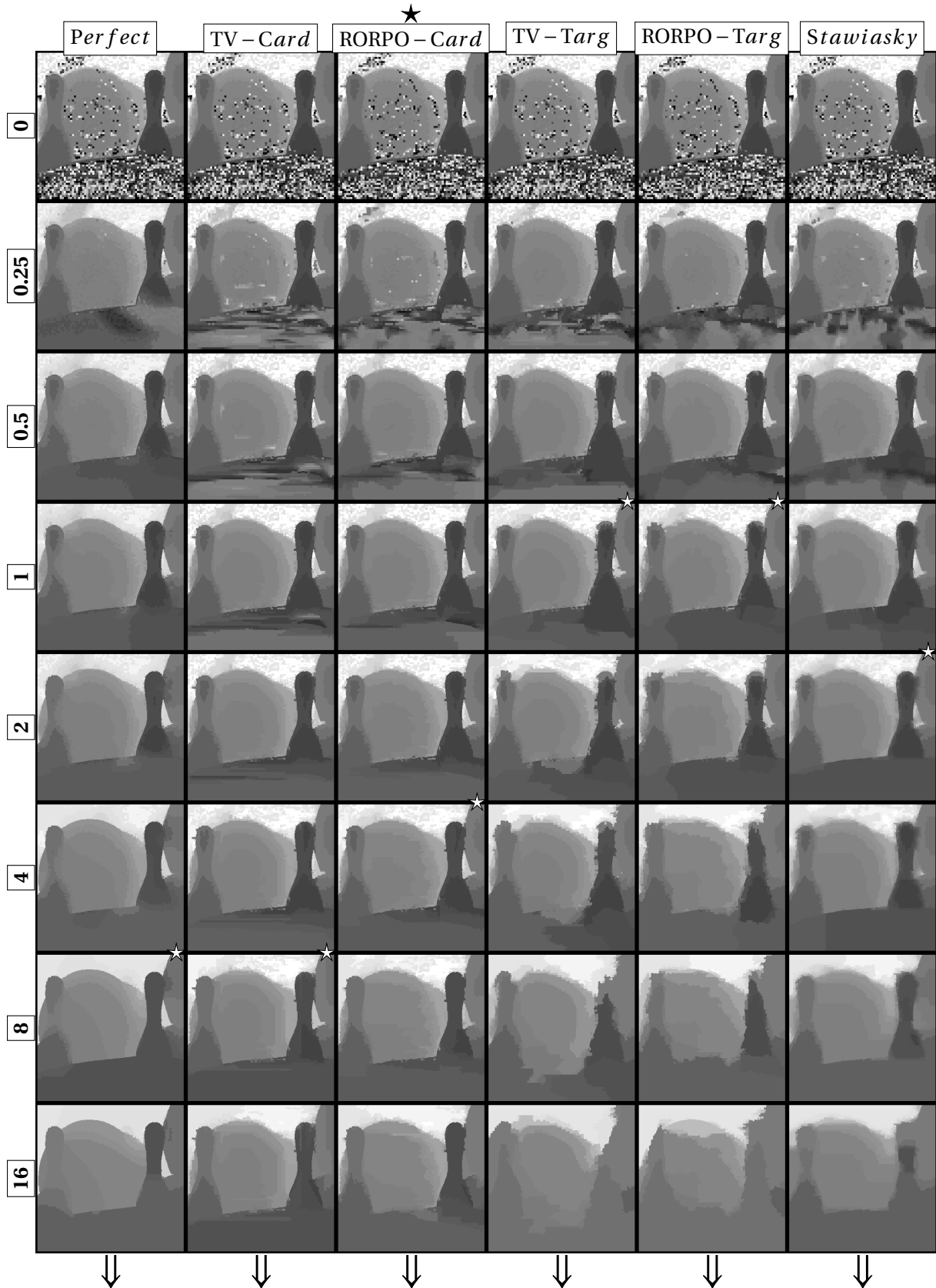


Figure 6.27 – Depth maps obtained with the scene *Bowling*, for the neighborhood strategies indicated above and multiple values of α indicated on the left hand side. For each neighborhood strategy, a star on the top right corner of a depth map indicates the best result according to PSNR. In this scene, the depth of the scene in the foreground is very badly estimated, and explains the bad result obtained with regularization. We note that despite the bad numerical results, the sharp pins are preserved when increasing the regularization with cardinal-based and perfect neighborhood, while they quickly deteriorate with other neighborhoods, which is an interesting property.

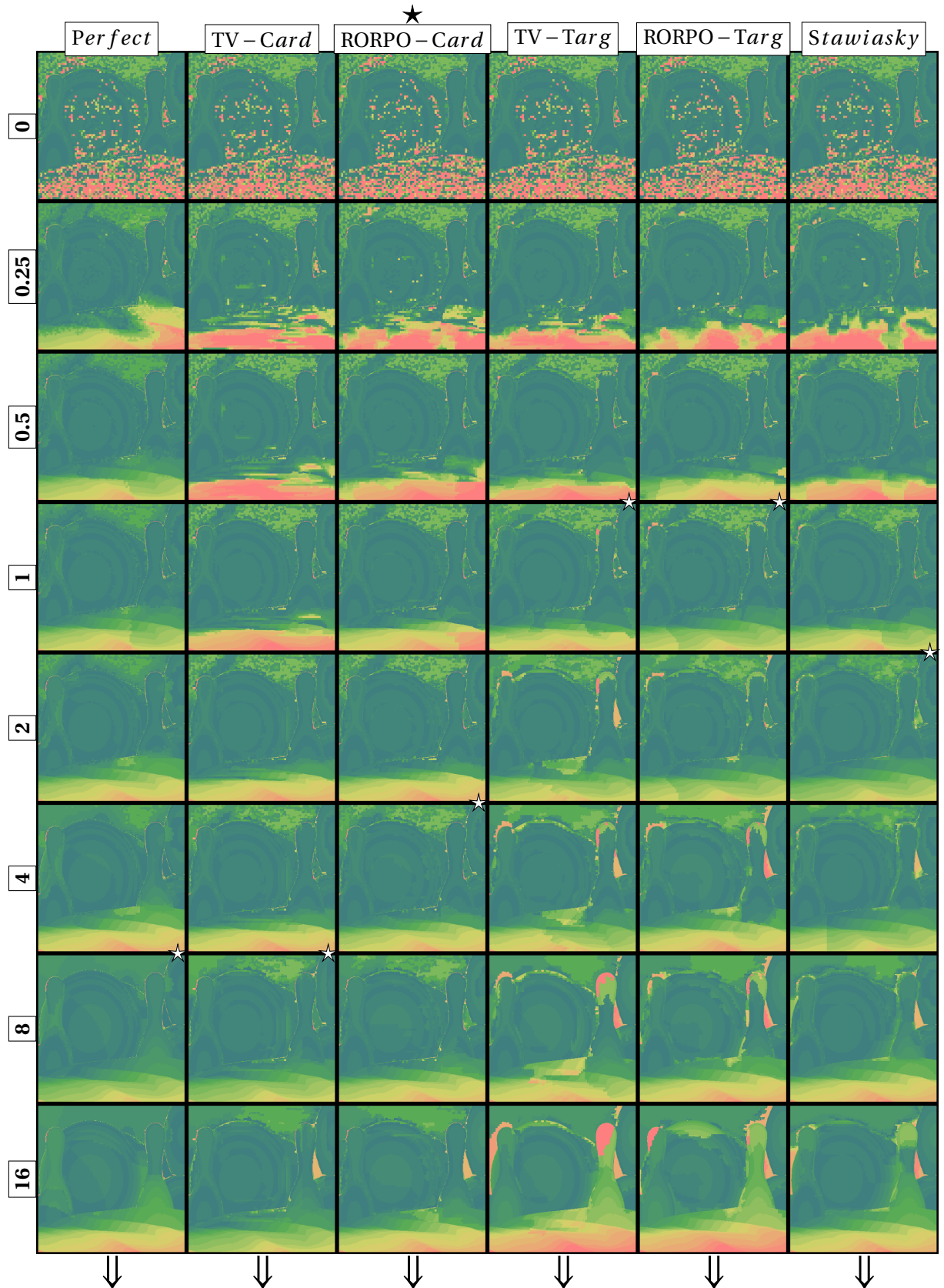


Figure 6.28 – Depth error maps obtained with the scene *Bowling*, for the neighborhood strategies indicated above and multiple values of α indicated at on the left hand side. For each neighborhood strategy, a star on the top right corner of a depth map indicates the best result according to PSNR. The main errors are brought by depth estimation error in the foreground and the background, and persist even when increasing α . Slight differences are visible between the two pins on the right hand side and near the sharp edges when regularization increases even more.

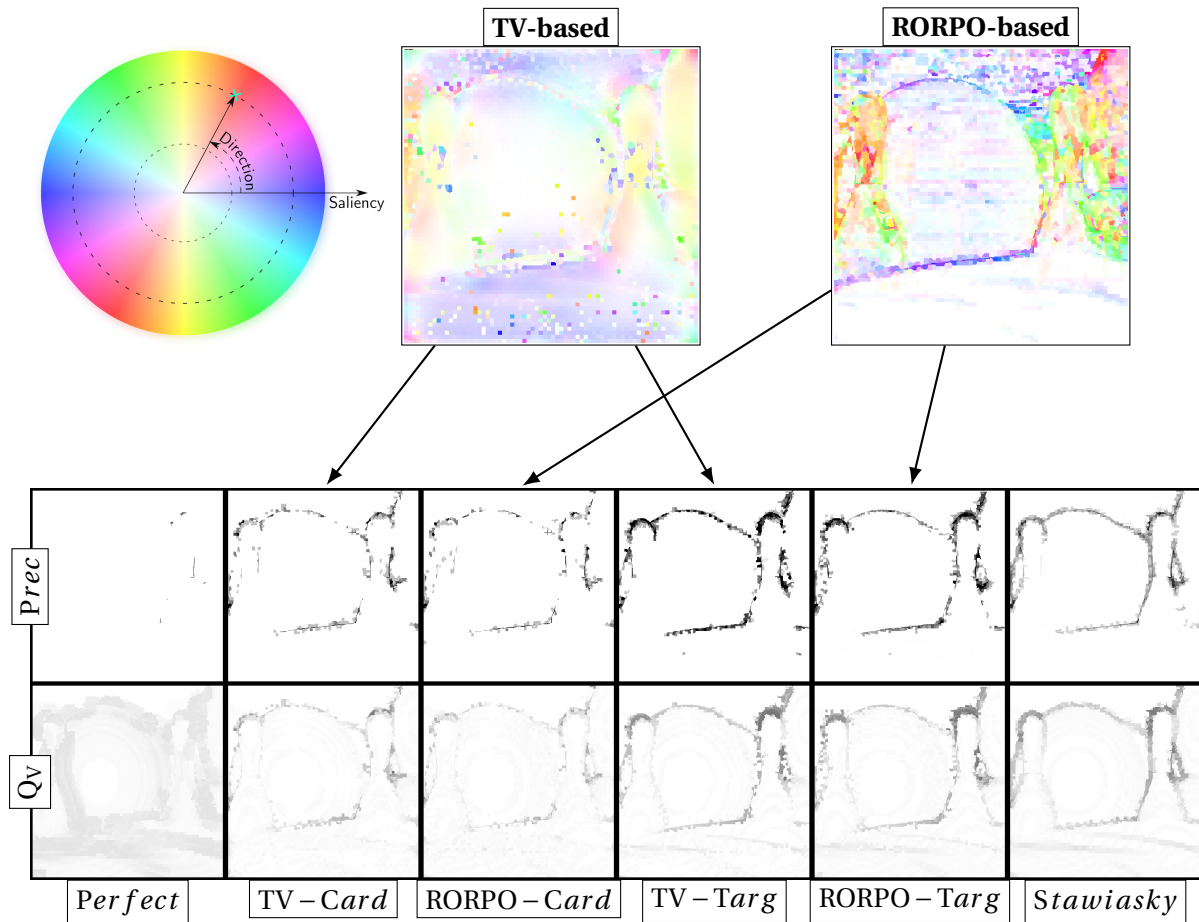


Figure 6.29 – Bowling neighborhood qualitative criteria displayed for the six neighborhood strategies. The guidance map computed from RORPO and TV clearly show the differences between smooth TV orientations with slight errors near the border of objects, and RORPO that highlights sharp areas but yield questionable orientations in the background, because of the regular pattern in the wall.

Art at Pixel level Since we have lead no exhaustive study about the impact of the number and nature of superpixels in our work so far, we bring here an experiment of our approach at pixel level. To some extent, pixels can be seen as a specific kind of superpixel, with the smallest possible size and a very regular lattice. Confronting our anisotropic neighborhoods to the case where sites are pixel is a way to highlight some limitations of our approach that can be the object of future works.

At pixel level, we consider the scene Art, since it has multiple thin structures and show mitigated results, see Figure 6.30.

In the foreground, the most prominent objects are a jug with a large handle, and a large brush, on the left hand side. On the right hand side, there is a small silhouette in modeling clay just below a cup, with small brushes in it. In the middle, we can see a huge pink grease pencil, below a conical pink shape. Finally, there is a carved head in the middle-ground, more small grease pencils in their box, and four hoops of varying sizes in the upper right corner of the image. In the background, some paints and posters are pinned to a white wall.

The brushes and the stems of the hoops are straight thin structures. The hoops, and the handle of the jug, are curved thin structures. On a larger scale, the area between the cone and the carved head can also be considered as a thin structure.

Figure 6.31 shows that the low sharpness and noisy areas are mostly the inner part of the jug, the areas in the background at the left and the right side of the cone, and the white wall section behind the hoops. The hoops themselves are also a lot prone to errors, as well as the end of the grease pencil. According to the three numerical criteria, the neighborhood that performs the best on that scene is [Stawiaski and Decenci re \[2011\]](#)'s one.

With every strategy, the stems of the hoops are hard to maintain when the regularization parameter grows, although when we look at RORPO cardinal-based neighborhood's best result, some parts of the hoops handle are still more distinguishable. The second main issue in the area of the hoops, is the wall's depth bad estimation, that require an important regularization. On this point, the behavior of each neighborhood is different, between hoops preservation and background's regularization, facing the challenges of over smoothing the results and inducing a bias. We can check this visually and in Figure 6.32 also.

If we look closer to the left side of the cone for instance, we can see the depth of the background in the isotropic neighborhood is badly estimated, while some of our approach estimate better the depth in that area.

When we look at quality maps for neighborhoods, in 6.33, we see as expected that the errors are concentrated on the edges of the objects, and that at pixel level, those edges are even smaller regions of the image. Even if we can find some small improvements, for instance on the top of the carved head, it is no surprise that the advantage of anisotropic neighborhoods diminishes at pixel level, while some of their flaws will impact the quality of the reconstruction.

One way to improve the results could be to adjust the anisotropic threshold Γ on the guidance map's saliency to ensure the anisotropic behavior of the neighborhoods is only encountered on the edges of the objects. Since we have, for computational reasons, computed the guidance map at superpixel level before deriving the neighborhood at pixel level, therefore, the guidance map may lack of precision, and we should reformulate some of these elements.



Figure 6.30 – The ground truth and the all-in-focus image for the scene *Art* of the dataset.

6.4.2 Sensitivity to the regularization parameter

In real life situations, the ground truth of a scene is not known. For having an estimation of the performances of the algorithm for a fixed value of λ , we have run tests over the dataset and summed the squared values of the RMSE of each scene, in order to approach the measure of the performance of the algorithm over the whole set of scenes. When the RMSE is the lowest, the best performances are achieved. The result is shown in Figure 6.34 at superpixel level and in Figure 6.35 at pixel level.

The proof of concept, namely the *perfect* neighborhood, outperforms all the others approaches at both pixel and superpixel level, for a wide range of regularization coefficients, the optimal choice seemingly being in the range $\alpha \in [1, 2]$. At pixel level, as expected, Stawiaski and Decenière [2011]’s neighborhood has better results than our approaches, which is visible in this dataset for $\alpha = 2$. Surprisingly, without having outstanding results, the RORPO-shape-based neighborhood may overcome our other neighborhoods for $\alpha = 1$ at pixel level, while its performances are below them at superpixel level.

However, at superpixel level, both RORPO and TV cardinal-based neighborhood yield good results for a large set of values of $\alpha \in [2, 16]$. This robustness of cardinal-based neighborhoods to the regularization parameter is also confirmed visually and is one of the strengths of this approach against the other implementations. In the same range of values of α , cardinal-based neighborhood also perform almost as well as the perfect neighborhood.

For questioning more aspects of the benefits of anisotropic neighborhood approaches with superpixels, we also compared the best results at superpixel and pixel level in Figure 6.36. An interesting fact is that working at superpixel level seems to improve the results of anisotropic neighborhoods, not only against isotropic superpixel neighborhood regularization, but also against isotropic neighborhood at pixel level. For instance, for $\alpha \in [4, 8]$, RORPO cardinal-based neighborhood outperforms all the other neighborhood constructions, excepted the *perfect* ones, for any regularization parameter value and for both pixel and superpixel level. This is an interesting perspective about the additional regularization that the superpixels may bring to the image that finally increases the average quality of the image.

6.4.3 Result tables

The exhaustive results obtained with best regularization parameter α for each scene, each neighborhood construction and each type of site (pixels and superpixels) are listed in Table 6.2, 6.3, 6.4

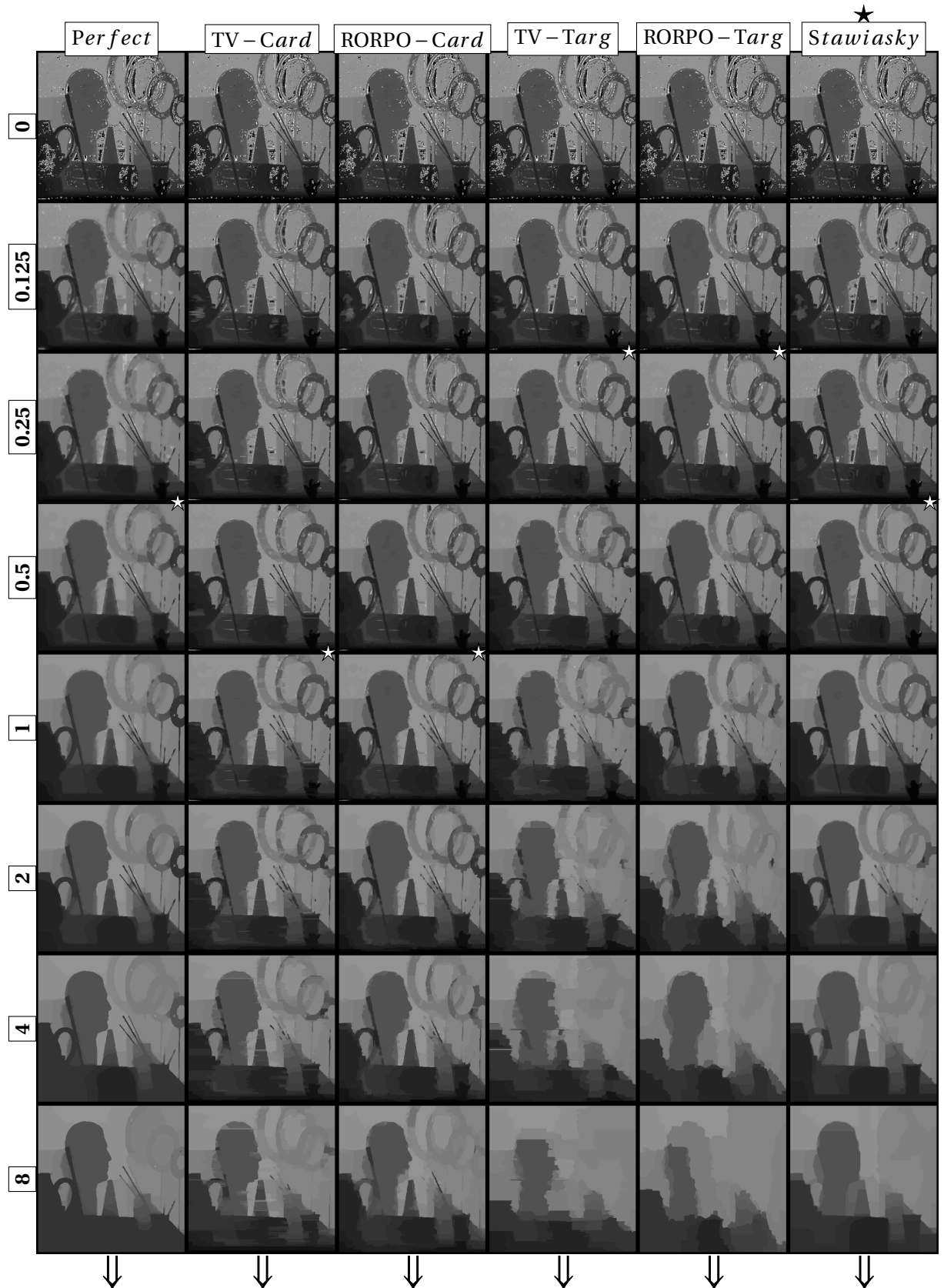


Figure 6.31 – Depth maps obtained with the scene *Art1*, at pixel level, for the neighborhood strategies indicated above and multiple values of α indicated on the left hand side. For each neighborhood strategy, a star on the top right corner of a depth map indicates the best result according to PSNR. In this scene, a few errors are concentrated on challenging areas, such as the hoops in the top right corner or the body of the jug. While the body of the jug and the grease pencil are quite well regularized, the main errors seem to occur near the hoops handles and the background behind the hoops.

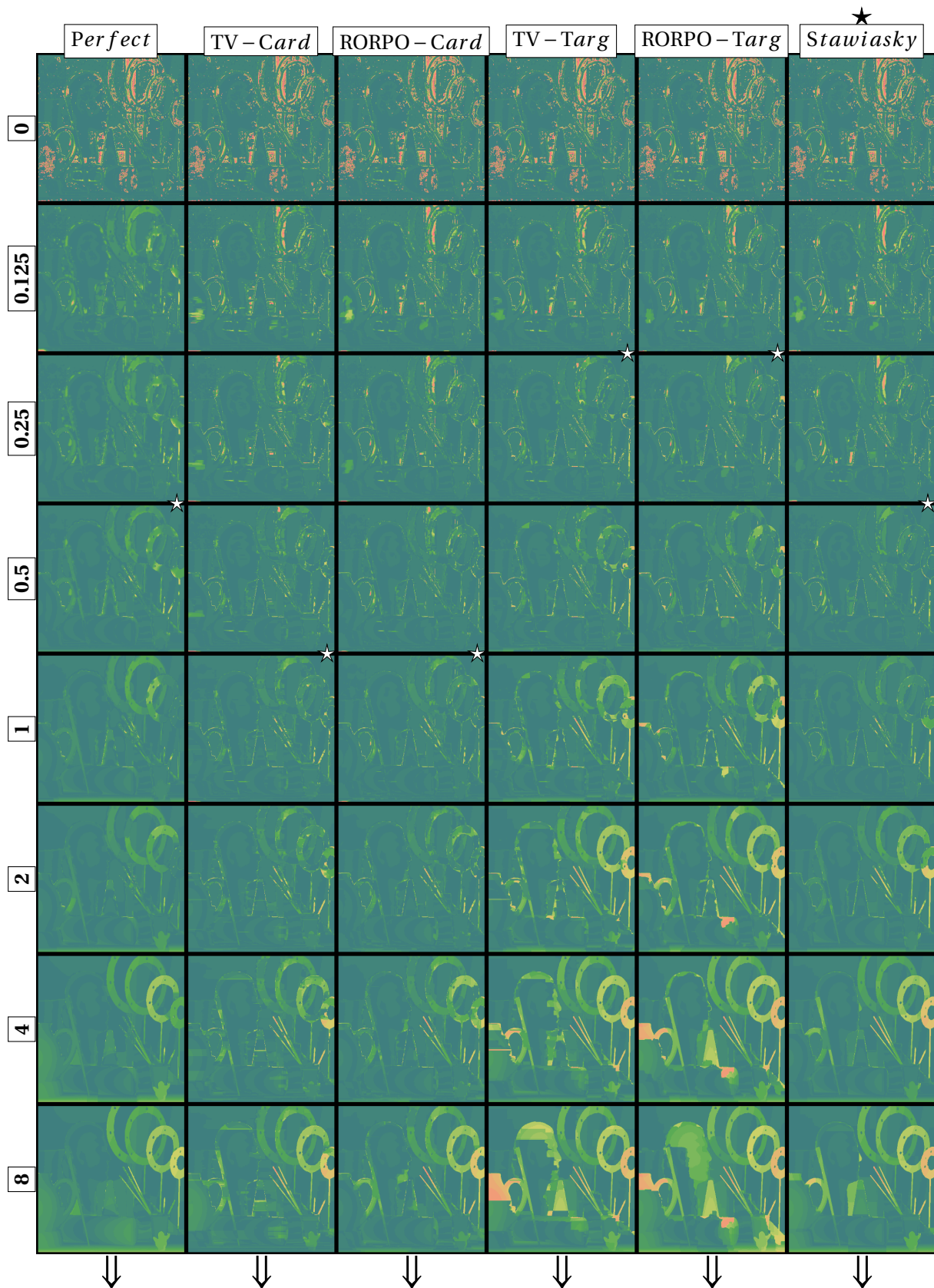


Figure 6.32 – Depth error maps obtained with the scene *Art1*, at pixel level, for the neighborhood strategies indicated above and multiple values of α indicated on the left hand side. For each neighborhood strategy, a star on the top right corner of a depth map indicates the best result according to PSNR. The remaining errors after regularization are quite contained, making visual estimation of the quality difficult, each neighborhood producing different results on the different areas of the scene.

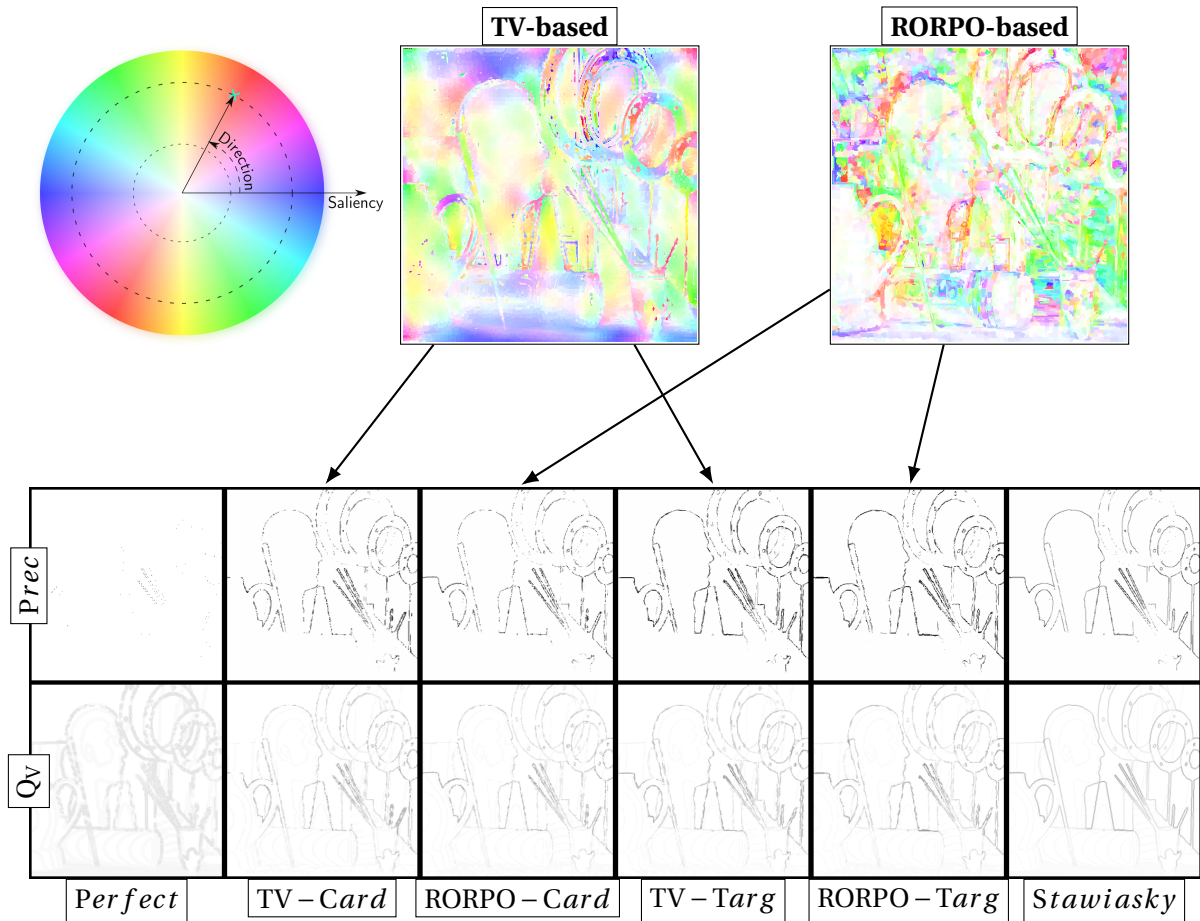


Figure 6.33 – Guidance map orientations for the *Art* scene computed from RORPO and TV neighborhood at pixel level. The Precision and quality criterion $Prec$ and Q_v are displayed for the six neighborhood on the second and third rows. At pixel level, the distinction between anisotropic and isotropic neighborhoods is quite difficult, since the edges of the objects where the error is concentrated become thinner, especially for Stawiaski and Decenci re [2011]’s neighborhood that becomes the 4-adjacency. Since the performances of the anisotropic neighborhood may be reduced in isotropic area by introducing erroneous patterns for instance, the results may even be degraded by such approaches at pixel level.

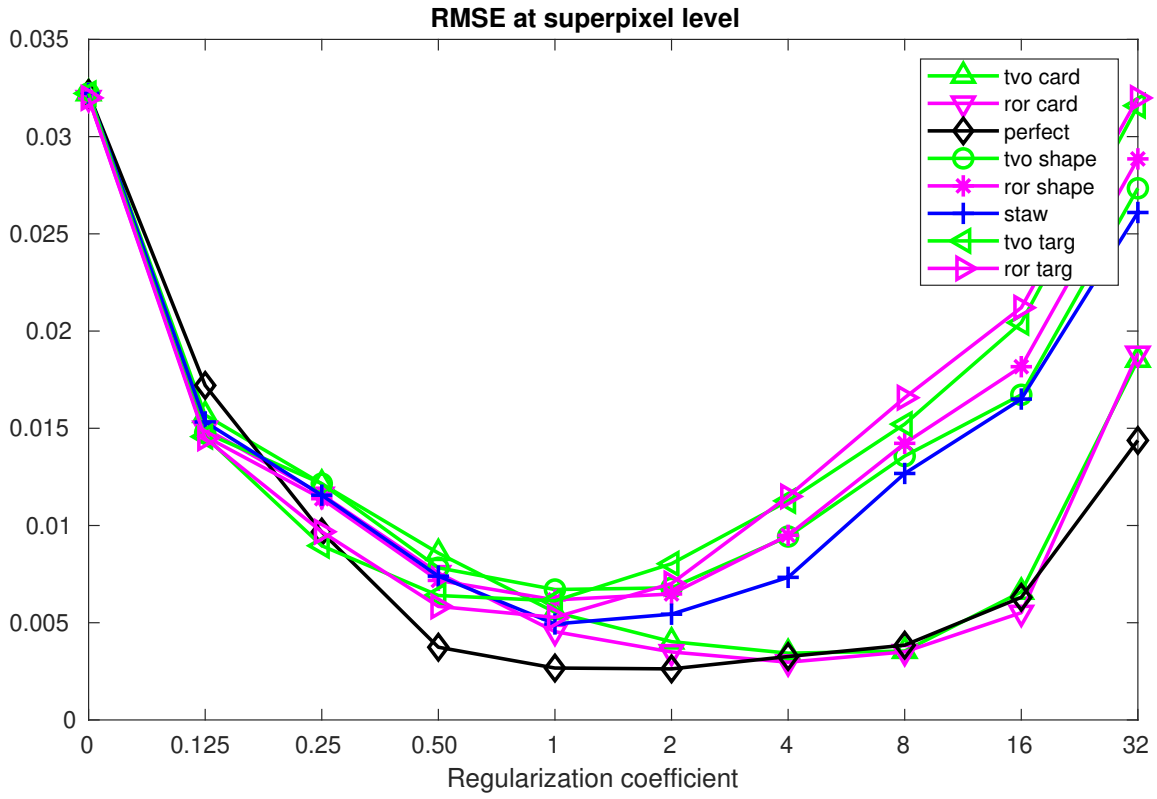


Figure 6.34 – RMSE at superpixel level on the whole dataset. Perfect and cardinal-based neighborhoods give the best results for a wider range of values of α than the other neighborhoods.

and 6.5. They represent the numerical values of the results obtained in Figure 6.10.

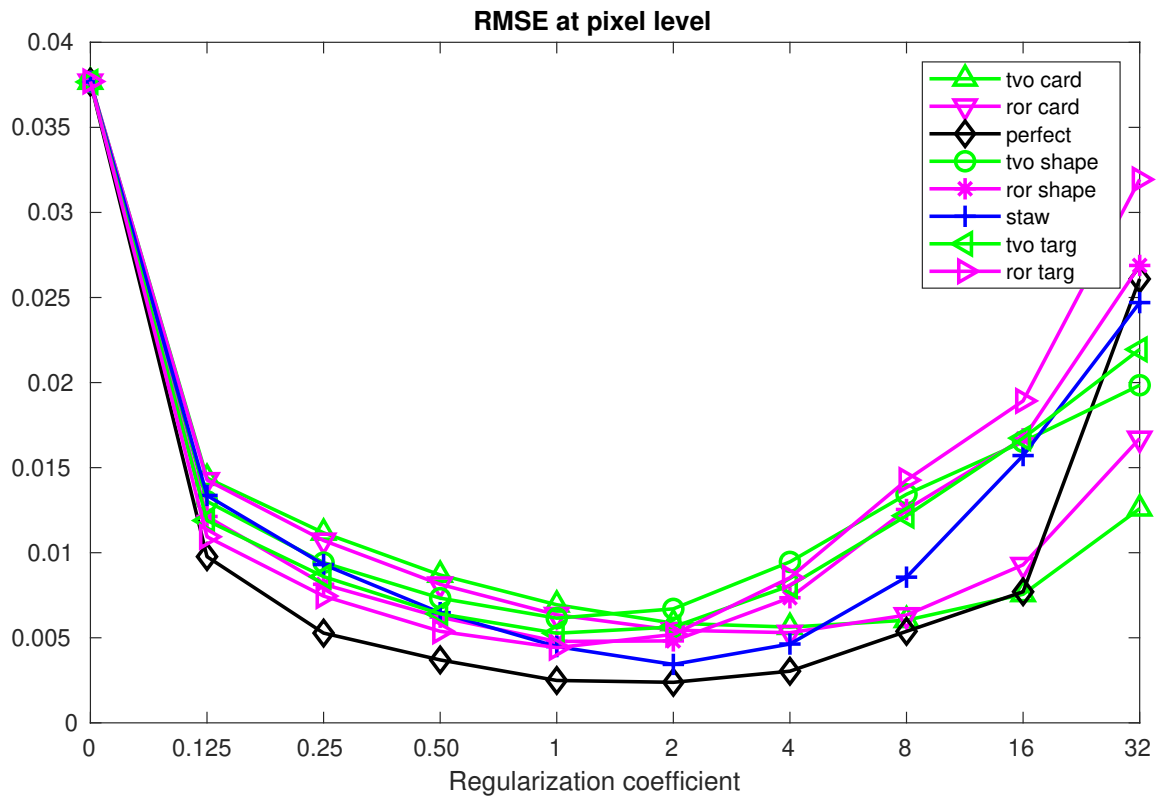


Figure 6.35 – RMSE at pixel level. When increasing the value of the regularization coefficient α , the RMSE error is reduced more rapidly than at superpixel level, but for $\alpha > 0.5$, the variations are generally smaller. The best results are however obtained with perfect and Stawiasky's neighborhood, and show the limits of anisotropic neighborhoods at pixel level.

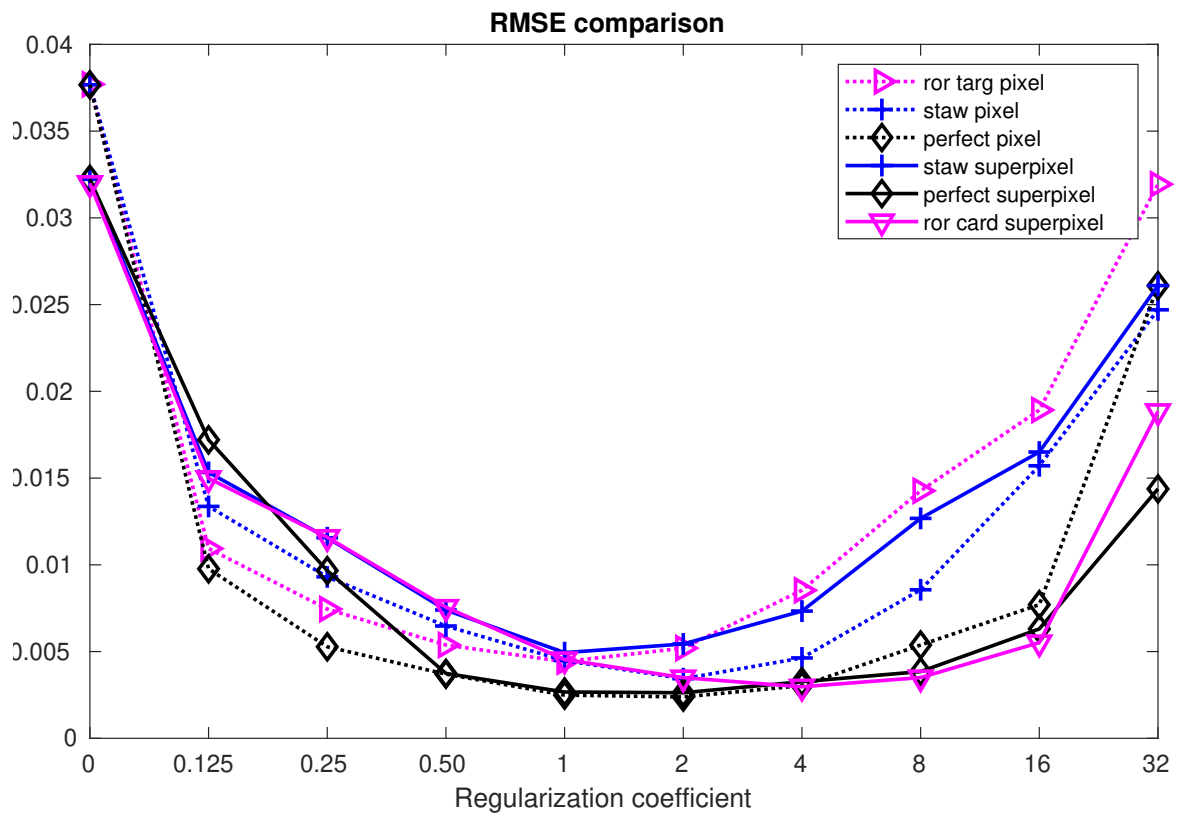


Figure 6.36 – Comparison of the mean RMSE values between pixel and superpixel level, for the best anisotropic neighborhood candidate of each and for Stawiasky and perfect neighborhoods. We note that pixel level neighborhood allows for a quick regularization in the beginning, but for higher values of α , superpixel level anisotropic neighborhood meets and exceeds the performances of pixel level neighborhoods, except for the perfect one.

| Nbh | | Perfect | TV-Card | RO-Card | TV-Targ | RO-Targ | TV-Shape | RO-Shape | Stawiasky |
|--------|-----|---------|---------|---------|---------|---------|----------|----------|-----------|
| Sample | | Art1 | | | | | | | |
| PSNR | pxl | 61.80 | 61.66 | 61.99 | 60.80 | 60.62 | 60.72 | 60.84 | 62.08 |
| | spx | 62.05 | 61.80 | 61.83 | 61.24 | 61.01 | 60.37 | 59.82 | 61.51 |
| UQI | pxl | 99.44 | 99.43 | 99.48 | 99.32 | 99.30 | 99.29 | 99.31 | 99.51 |
| | spx | 99.50 | 99.48 | 99.48 | 99.40 | 99.35 | 99.24 | 99.12 | 99.43 |
| SSIM | pxl | 98.87 | 99.08 | 99.14 | 99.06 | 98.94 | 98.90 | 98.97 | 99.21 |
| | spx | 98.95 | 99.15 | 99.19 | 98.99 | 98.83 | 98.71 | 98.45 | 99.04 |
| Sample | | Brush1 | | | | | | | |
| PSNR | pxl | 64.51 | 60.46 | 61.06 | 62.19 | 61.59 | 60.94 | 61.05 | 63.74 |
| | spx | 63.85 | 62.75 | 62.87 | 60.17 | 60.57 | 59.64 | 59.74 | 61.13 |
| UQI | pxl | 99.75 | 99.49 | 99.55 | 99.65 | 99.61 | 99.55 | 99.56 | 99.72 |
| | spx | 99.73 | 99.68 | 99.68 | 99.49 | 99.52 | 99.42 | 99.44 | 99.58 |
| SSIM | pxl | 96.61 | 96.91 | 97.25 | 97.4 | 97.45 | 97.02 | 97.38 | 98.21 |
| | spx | 95.24 | 97.67 | 97.38 | 96.22 | 96.22 | 95.5 | 95.15 | 96.39 |
| Sample | | Brush2 | | | | | | | |
| PSNR | pxl | 64.01 | 60.7 | 61.32 | 62.18 | 61.68 | 60.89 | 61.2 | 64.07 |
| | spx | 63.14 | 62.62 | 63.04 | 60.26 | 60.98 | 59.63 | 60.29 | 61.47 |
| UQI | pxl | 99.71 | 99.5 | 99.55 | 99.63 | 99.6 | 99.53 | 99.56 | 99.72 |
| | spx | 99.66 | 99.65 | 99.67 | 99.48 | 99.54 | 99.39 | 99.46 | 99.58 |
| SSIM | pxl | 96.64 | 96.82 | 97.35 | 97.39 | 97.45 | 97.06 | 97.4 | 98.05 |
| | spx | 95.08 | 97.66 | 97.52 | 96.16 | 96.57 | 95.35 | 95.65 | 96.46 |
| Sample | | Brush2c | | | | | | | |
| PSNR | pxl | 63.81 | 60.28 | 60.99 | 61.63 | 61.36 | 60.74 | 61.11 | 63.52 |
| | spx | 62.76 | 61.22 | 61.1 | 59.01 | 59.7 | 58.7 | 58.81 | 60.59 |
| UQI | pxl | 99.69 | 99.44 | 99.5 | 99.58 | 99.55 | 99.5 | 99.53 | 99.69 |
| | spx | 99.65 | 99.54 | 99.52 | 99.33 | 99.39 | 99.29 | 99.26 | 99.49 |
| SSIM | pxl | 96.57 | 96.81 | 97.39 | 97.28 | 97.43 | 97.04 | 97.38 | 98.15 |
| | spx | 94.85 | 97.22 | 97.25 | 95.95 | 96.2 | 95.3 | 95.2 | 96.31 |
| Sample | | Aloe1 | | | | | | | |
| PSNR | pxl | 63.43 | 63.46 | 63.43 | 63.03 | 62.97 | 63.23 | 63.28 | 63.58 |
| | spx | 62.49 | 62.41 | 62.4 | 62.29 | 62.28 | 62.29 | 62.21 | 62.29 |
| UQI | pxl | 99.85 | 99.86 | 99.86 | 99.84 | 99.84 | 99.85 | 99.85 | 99.86 |
| | spx | 99.82 | 99.82 | 99.82 | 99.81 | 99.81 | 99.81 | 99.81 | 99.81 |
| SSIM | pxl | 99.77 | 99.84 | 99.83 | 99.81 | 99.79 | 99.82 | 99.81 | 99.84 |
| | spx | 99.76 | 99.78 | 99.78 | 99.75 | 99.75 | 99.75 | 99.74 | 99.75 |
| Nbh | | Perfect | TV-Card | RO-Card | TV-Targ | RO-Targ | TV-Shape | RO-Shape | Stawiasky |

Table 6.2 – Global Performance Table 1: In this table we indicate the best results of each scene with respect to the regularization coefficient α .

| Nbh | | Perfect | TV-Card | RO-Card | TV-Targ | RO-Targ | TV-Shape | RO-Shape | Stawiasky |
|--------|-----|-----------|---------|---------|---------|---------|----------|----------|-----------|
| Sample | | Baby11 | | | | | | | |
| PSNR | pxl | 64.61 | 61.68 | 63.66 | 61.76 | 62.18 | 61.88 | 64.14 | 65.35 |
| | spx | 63.14 | 62.94 | 63.13 | 61.38 | 61.1 | 62.83 | 62.99 | 62.94 |
| UQI | pxl | 99.36 | 98.44 | 99.19 | 98.6 | 98.81 | 98.51 | 99.27 | 99.44 |
| | spx | 98.85 | 98.95 | 98.95 | 98.37 | 98.31 | 98.94 | 98.96 | 98.98 |
| SSIM | pxl | 93.28 | 87.85 | 93.46 | 90.87 | 92.13 | 90.47 | 94.78 | 97.12 |
| | spx | 88.94 | 91.53 | 89.95 | 88.5 | 89.01 | 91.73 | 92.44 | 92.62 |
| Sample | | Books1 | | | | | | | |
| PSNR | pxl | 67.68 | 64.79 | 64.7 | 64.45 | 64.2 | 64.71 | 66.1 | 67.07 |
| | spx | 67.33 | 67.07 | 67.1 | 66.21 | 66.35 | 67.01 | 66.95 | 67.19 |
| UQI | pxl | 99.9 | 99.74 | 99.76 | 99.74 | 99.71 | 99.76 | 99.83 | 99.88 |
| | spx | 99.9 | 99.89 | 99.89 | 99.85 | 99.86 | 99.88 | 99.88 | 99.89 |
| SSIM | pxl | 99.22 | 98.9 | 99.02 | 99.04 | 99.01 | 98.88 | 99.15 | 99.34 |
| | spx | 99.35 | 99.55 | 99.57 | 99.44 | 99.47 | 99.55 | 99.52 | 99.54 |
| Sample | | Bowling11 | | | | | | | |
| PSNR | pxl | 54.6 | 50.66 | 52.38 | 52.73 | 54.34 | 52.42 | 53.58 | 54.28 |
| | spx | 54.14 | 53.58 | 52.56 | 52.59 | 53.34 | 52.45 | 53.74 | 53.84 |
| UQI | pxl | 97.73 | 94.89 | 96.35 | 96.65 | 97.6 | 96.42 | 97.18 | 97.59 |
| | spx | 97.4 | 97.07 | 96.43 | 96.51 | 96.99 | 96.48 | 97.26 | 97.33 |
| SSIM | pxl | 90.66 | 88.94 | 88.75 | 89.96 | 90.89 | 89.01 | 89.69 | 90.66 |
| | spx | 89.69 | 89.73 | 88.69 | 89.22 | 89.53 | 88.65 | 89.74 | 89.81 |
| Sample | | Cloths11 | | | | | | | |
| PSNR | pxl | 71.11 | 71.13 | 71.13 | 71.12 | 71.12 | 71.14 | 71.14 | 71.15 |
| | spx | 70.44 | 70.44 | 70.44 | 70.46 | 70.44 | 70.44 | 70.45 | 70.45 |
| UQI | pxl | 99.88 | 99.88 | 99.88 | 99.88 | 99.88 | 99.88 | 99.88 | 99.88 |
| | spx | 99.85 | 99.85 | 99.85 | 99.85 | 99.85 | 99.85 | 99.85 | 99.85 |
| SSIM | pxl | 99.53 | 99.53 | 99.53 | 99.53 | 99.53 | 99.53 | 99.53 | 99.53 |
| | spx | 99.5 | 99.51 | 99.51 | 99.51 | 99.5 | 99.5 | 99.5 | 99.5 |
| Sample | | Dolls1 | | | | | | | |
| PSNR | pxl | 66.4 | 64.96 | 65.34 | 64.47 | 64.9 | 65.34 | 65.44 | 68.12 |
| | spx | 67.08 | 67.72 | 68.15 | 66.52 | 66.51 | 66.09 | 66.16 | 67.65 |
| UQI | pxl | 99.86 | 99.75 | 99.78 | 99.71 | 99.74 | 99.78 | 99.78 | 99.93 |
| | spx | 99.9 | 99.92 | 99.94 | 99.86 | 99.86 | 99.83 | 99.84 | 99.92 |
| SSIM | pxl | 98.98 | 98.34 | 98.48 | 98.26 | 98.62 | 98.39 | 98.59 | 99.19 |
| | spx | 98.95 | 99.13 | 99.22 | 98.8 | 98.75 | 98.59 | 98.64 | 99.09 |
| Nbh | | Perfect | TV-Card | RO-Card | TV-Targ | RO-Targ | TV-Shape | RO-Shape | Stawiasky |

Table 6.3 – Global Performance Table 2: In this table we indicate the best results of each scene with respect to the regularization coefficient α .

| Nbh | | Perfect | TV-Card | RO-Card | TV-Targ | RO-Targ | TV-Shape | RO-Shape | Stawiasky |
|--------|-----|-------------|---------|---------|---------|---------|----------|----------|-----------|
| Sample | | Flowerpots1 | | | | | | | |
| PSNR | pxl | 60.87 | 56.65 | 59.13 | 57.58 | 60.04 | 58.05 | 60.29 | 60.37 |
| | spx | 60.41 | 61.27 | 61.75 | 60.53 | 60.16 | 60.56 | 60.39 | 60.72 |
| UQI | pxl | 95.87 | 89.6 | 93.95 | 91.5 | 95.06 | 92.26 | 95.4 | 95.4 |
| | spx | 95.44 | 96.26 | 96.75 | 95.72 | 95.3 | 95.8 | 95.53 | 95.85 |
| SSIM | pxl | 90.35 | 84.41 | 89.2 | 85.34 | 89.08 | 85.02 | 90.07 | 89.95 |
| | spx | 89.33 | 90.83 | 92.31 | 90.37 | 89.98 | 91.11 | 90.35 | 90.6 |
| Sample | | Lampshade11 | | | | | | | |
| PSNR | pxl | 56.96 | 53.32 | 53.32 | 53.09 | 53.25 | 52.39 | 53.27 | 54.61 |
| | spx | 56.82 | 56.65 | 57.78 | 53.76 | 55.13 | 53.29 | 55.76 | 54.15 |
| UQI | pxl | 97.78 | 95.23 | 95.19 | 94.93 | 95.12 | 94.1 | 95.11 | 96.34 |
| | spx | 97.75 | 97.63 | 98.18 | 95.67 | 96.77 | 95.25 | 97.13 | 95.94 |
| SSIM | pxl | 85.21 | 83.43 | 83 | 83.34 | 83.46 | 82.56 | 82.83 | 83.81 |
| | spx | 85.13 | 85.05 | 86.33 | 83.02 | 84.53 | 83.05 | 84.4 | 83.6 |
| Sample | | Laundry1 | | | | | | | |
| PSNR | pxl | 65.46 | 64.08 | 63.95 | 63.51 | 62.94 | 63.86 | 64.34 | 64.77 |
| | spx | 64.64 | 64.79 | 64.56 | 62.91 | 63.2 | 63.24 | 63.04 | 64.15 |
| UQI | pxl | 99.86 | 99.78 | 99.76 | 99.75 | 99.7 | 99.77 | 99.8 | 99.82 |
| | spx | 99.84 | 99.83 | 99.82 | 99.7 | 99.73 | 99.72 | 99.71 | 99.79 |
| SSIM | pxl | 99.43 | 99.4 | 99.45 | 99.38 | 99.2 | 99.38 | 99.4 | 99.36 |
| | spx | 99.42 | 99.52 | 99.47 | 99.21 | 99.26 | 99.22 | 99.17 | 99.36 |
| Sample | | Midd11 | | | | | | | |
| PSNR | pxl | 65.86 | 61.6 | 66.32 | 61.39 | 65.87 | 62.03 | 65.87 | 66.22 |
| | spx | 64.54 | 65.81 | 65.75 | 63.76 | 63.76 | 61.54 | 62.19 | 64.53 |
| UQI | pxl | 99.88 | 99.62 | 99.9 | 99.59 | 99.88 | 99.66 | 99.88 | 99.89 |
| | spx | 99.84 | 99.89 | 99.89 | 99.79 | 99.79 | 99.56 | 99.65 | 99.83 |
| SSIM | pxl | 98.9 | 98.44 | 98.97 | 98.48 | 98.92 | 98.45 | 98.92 | 98.95 |
| | spx | 98.74 | 99.11 | 99.02 | 98.76 | 98.72 | 98.51 | 98.67 | 98.9 |
| Sample | | Moebius1 | | | | | | | |
| PSNR | pxl | 66.13 | 63.93 | 65.97 | 63.82 | 64.3 | 64 | 64.97 | 66.11 |
| | spx | 66.1 | 66.56 | 66.83 | 65.35 | 64.54 | 64.81 | 63.93 | 65.5 |
| UQI | pxl | 99.85 | 99.73 | 99.84 | 99.72 | 99.75 | 99.73 | 99.79 | 99.85 |
| | spx | 99.86 | 99.87 | 99.88 | 99.82 | 99.77 | 99.79 | 99.74 | 99.82 |
| SSIM | pxl | 98.73 | 98.67 | 98.93 | 98.63 | 98.65 | 98.62 | 98.71 | 98.98 |
| | spx | 98.91 | 99.12 | 99.28 | 98.86 | 98.73 | 98.7 | 98.46 | 98.89 |
| Nbh | | Perfect | TV-Card | RO-Card | TV-Targ | RO-Targ | TV-Shape | RO-Shape | Stawiasky |

Table 6.4 – Global Performance Table 3: In this table we indicate the best results of each scene with respect to the regularization coefficient α .

| Nbh | | Perfect | TV-Card | RO-Card | TV-Targ | RO-Targ | TV-Shape | RO-Shape | Stawiasky |
|--------|-----|-----------|---------|---------|---------|---------|----------|----------|-----------|
| Sample | | Monopoly1 | | | | | | | |
| PSNR | pxl | 67.26 | 62.51 | 64.33 | 62.17 | 64.04 | 62.84 | 64.55 | 65.57 |
| | spx | 64.18 | 61.88 | 61.9 | 61.67 | 59.94 | 61.58 | 60.21 | 60.72 |
| UQI | pxl | 99.88 | 99.55 | 99.72 | 99.5 | 99.7 | 99.58 | 99.73 | 99.79 |
| | spx | 99.65 | 99.43 | 99.4 | 99.37 | 98.98 | 99.29 | 99.02 | 99.14 |
| SSIM | pxl | 99.26 | 99.02 | 99.21 | 99.04 | 99.14 | 99.07 | 99.23 | 99.24 |
| | spx | 98.98 | 98.93 | 98.94 | 98.76 | 98.47 | 98.76 | 98.51 | 98.65 |
| Sample | | Plastic1 | | | | | | | |
| PSNR | pxl | 53.91 | 49.67 | 50.43 | 51.39 | 53.28 | 52.54 | 53.34 | 52.98 |
| | spx | 55.42 | 53.42 | 53.71 | 51.45 | 52.73 | 51.82 | 52.33 | 52.62 |
| UQI | pxl | 92.54 | 83.29 | 85.35 | 87.63 | 91.61 | 90.12 | 91.75 | 90.98 |
| | spx | 94.52 | 91.62 | 92.28 | 87.7 | 90.62 | 88.55 | 89.81 | 90.2 |
| SSIM | pxl | 78.77 | 73.97 | 73.97 | 74.62 | 77.69 | 76.48 | 77.84 | 77 |
| | spx | 81.62 | 78.28 | 80.05 | 74.9 | 77.07 | 75.03 | 76.37 | 76.55 |
| Sample | | Reindeer1 | | | | | | | |
| PSNR | pxl | 65.66 | 62.5 | 64.62 | 60.47 | 61.63 | 60.63 | 62.11 | 63.94 |
| | spx | 66.28 | 64.9 | 63.83 | 60.78 | 61.01 | 59.42 | 59.13 | 61.59 |
| UQI | pxl | 99.76 | 99.44 | 99.68 | 99.03 | 99.27 | 99.03 | 99.35 | 99.61 |
| | spx | 99.83 | 99.74 | 99.62 | 99.16 | 99.18 | 98.75 | 98.67 | 99.28 |
| SSIM | pxl | 96.79 | 96.13 | 97.18 | 95.06 | 95.32 | 94.28 | 95.39 | 96.4 |
| | spx | 98.16 | 98.57 | 97.89 | 96.49 | 96.05 | 95.82 | 94.14 | 95.53 |
| Sample | | Rocks11 | | | | | | | |
| PSNR | pxl | 70.05 | 69.06 | 69.52 | 68.97 | 69.27 | 69.49 | 69.9 | 70.1 |
| | spx | 69.72 | 69.49 | 69.28 | 69.64 | 69.68 | 69.57 | 69.69 | 69.76 |
| UQI | pxl | 99.85 | 99.81 | 99.83 | 99.8 | 99.81 | 99.83 | 99.84 | 99.85 |
| | spx | 99.84 | 99.83 | 99.82 | 99.84 | 99.84 | 99.83 | 99.84 | 99.84 |
| SSIM | pxl | 99.39 | 99.38 | 99.43 | 99.35 | 99.4 | 99.33 | 99.43 | 99.42 |
| | spx | 99.44 | 99.41 | 99.42 | 99.38 | 99.44 | 99.4 | 99.43 | 99.42 |
| Sample | | Wood11 | | | | | | | |
| PSNR | pxl | 67.35 | 66.17 | 66.26 | 64.92 | 65.03 | 66.18 | 66.36 | 67.41 |
| | spx | 67 | 66.74 | 66.61 | 66.72 | 65.23 | 66.49 | 66.16 | 66.39 |
| UQI | pxl | 99.88 | 99.83 | 99.84 | 99.76 | 99.77 | 99.83 | 99.84 | 99.88 |
| | spx | 99.87 | 99.86 | 99.85 | 99.86 | 99.78 | 99.85 | 99.83 | 99.84 |
| SSIM | pxl | 99.55 | 99.54 | 99.63 | 99.41 | 99.45 | 99.47 | 99.55 | 99.63 |
| | spx | 99.36 | 99.63 | 99.62 | 99.55 | 99.26 | 99.49 | 99.37 | 99.35 |
| Sample | | Wood21 | | | | | | | |
| PSNR | pxl | 60.29 | 54.54 | 57.45 | 52.61 | 59.56 | 51.73 | 60.23 | 60.51 |
| | spx | 56.35 | 55.39 | 57.53 | 54.71 | 56.78 | 55.04 | 57.27 | 57.26 |
| UQI | pxl | 98.72 | 95.27 | 97.35 | 93.07 | 98.49 | 91.62 | 98.73 | 98.81 |
| | spx | 96.41 | 95.82 | 97.42 | 95.27 | 96.99 | 95.44 | 97.3 | 97.29 |
| SSIM | pxl | 90.05 | 86.55 | 86.21 | 86.52 | 89.89 | 85.24 | 90.11 | 90.18 |
| | spx | 84.2 | 84.45 | 86.3 | 85.05 | 86.66 | 83.94 | 86.82 | 86.55 |
| Nbh | | Perfect | TV-Card | RO-Card | TV-Targ | RO-Targ | TV-Shape | RO-Shape | Stawiasky |

Table 6.5 – Global Performance Table 4: In this table we indicate the best results of each scene with respect to the regularization coefficient α .

6.5 Conclusion

In this chapter, we have implemented two approaches for the detection of thin structures on SFF, namely TV and RORPO. For both methods, we have constructed three types of anisotropic neighborhoods, namely shape-based, cardinal-based and target-based. Experiments have been conducted both at pixel and superpixel level, for evaluating the benefits of using anisotropic neighborhoods against isotropic ones. Complementary information also have been gathered during the redaction of an article, which extracts are in the Appendix C, and that include computation times and some experiments about the effect of the choice of the algorithm of superpixels.

The results are clearly encouraging at superpixel level, since on the average, RORPO-cardinal-based neighborhood seems to produce consistently better results than isotropic approaches proposed by [Stawiaski and Decenci re \[2011\]](#). While at pixel level, isotropic approach of [Stawiaski and Decenci re \[2011\]](#) mostly outperforms our anisotropic neighborhoods, it is worth noting that the RMSE performances of RORPO-cardinal-based neighborhood at superpixel are still better than [Stawiaski and Decenci re \[2011\]](#)’s results at pixel level, for a reasonable computing time.

Based on these observations, anisotropic superpixel neighborhoods seem to offer promising perspectives. For instance, we can imagine conducting systematic experiments on larger dataset, for ensuring the observed trend. Since the parameters have only been tuned by hand, at multiple stages of maturation of the thesis, we can also imagine a more systematic study of the effect of each set of parameter. Using techniques from other fields such as deep learning could also allow be beneficial to automatically learn the optimal parameters, depending on the data.

Many perspectives of improvement also have been mentioned when listing the weakness of each method proposed. To pick an example, the guidance map produced by TV could be derived from the selection of tensors from a slightly regularized depth map, instead of a blind depth map, for avoiding outliers in the guidance map. We could compute the algorithm differently so that the guidance map could also be computed at pixel level.

There is therefore room for improvement of these approaches without risking any over-fitting of the application. Still, the results presented here prove the positive contribution of anisotropic regularization for multiple scenes.

Chapter 7

Conclusions and perspectives

7.1 Conclusions

In this thesis, we take up the challenge of defining anisotropic neighborhoods with generic definition with respect to the elements of an image, that are either pixels or superpixels. The objective of such neighborhoods was to allow for the preservation of thin structures while regularizing the segmentation of an image.

For doing so, we propose three methods for segmenting the thin structures on the images, and then implement four anisotropic neighborhood constructions based onto the derived guidance map. Two applications were considered for evaluating the proposed approaches, namely binary segmentation with thin structures and Shape From Focus.

In the case of the binary segmentation application, we have achieved many experiments with the energy-based estimation of thin structures combined with the dictionary-based neighborhood, for a large number of shapes (ellipses, rectangles, cones), of different sizes and for different numbers of orientations. In practice, this approach has shown some limits in terms of robustness to labeling errors, as well as some artifacts near the structures of the images.

This led us to consider the Tensor Voting-based approach, associated with shape and path-based neighborhoods. Visually, Tensor Voting seems to yield more convincing and smoother guidance map than energy-based approach. With the simulated image, when it comes to numerical comparisons of the neighborhood constructions, the F-measure criterion indicates better performance for the cardinal-based neighborhood than for the other approaches, including [Stawiaski and Decencière \[2011\]](#)'s, proposed as the isotropic reference. However, the results are very close, and the trend is reversing on the crack image. This effect is likely to partially come from the texture of crack images, that produces irregular superpixels.

We therefore propose a more comprehensive study in the Shape From Focus framework, including Tensor Voting and RORPO-based guidance maps and all the proposed anisotropic neighborhoods, except the dictionary-based one. On most scenes, the conclusions regarding the guidance map constructions are the same: Tensor Voting yield smoother guidance maps, when the initial depth is rather well estimated, while RORPO yields contrasted guidance maps, able to better preserve sharp details of the image.

Despite the fact that, at pixel level, the benefit of anisotropic neighborhood (with respect to [Stawiaski and Decencière \[2011\]](#)) appears mitigated, it appears clearly that RORPO-cardinal-based neighborhood at superpixel level outperforms the other competitors, with robust performance with respect to the regularization parameter α . The qualitative comparison shows cardinal-based neighborhood allows for better preservation of thin structures even with higher values of α . Furthermore, the comparison of pixel and superpixel level performance shows that segmentation obtained with RORPO-cardinal-based neighborhood still surprisingly outperforms segmentation obtained with [Stawiaski and Decencière \[2011\]](#)'s neighborhood at pixel level.

7.2 Perspectives

These results are therefore promising while raising issues that allow us to consider interesting perspectives for future work. Indeed, many possible points of the study could not be addressed, due to limited time. In particular, we could evoke the impact of the choice and the number of superpixels, and their preprocessing. The question of the regularization of depth maps could also be raised, especially in the case of RORPO, that is faithful on the edges on the objects, but may be slightly noisy in their center or when the sharpness is low. The set of parameters to build the map of thin structures, whose ground truth is unknown, such as the length of the opening of the paths, or the parameters of the Tensor Voting, have been set empirically after qualitative observations, but a thorough study could have been conducted. At the level of the construction of the neighborhoods, the same questions arise, the number of parameters playing on the quality of the neighborhoods being rather large.

It could also be interesting for instance to get rid of the effect of the energy data term when we know that some observations are not “reliable”. This would ensure convergence of such areas to consistent values, and solve some issues we had in some scenes considered in Shape From Focus application. Moreover, partially disabling the data term rises an interesting question about the neighborhoods and the cliques. Since the regularity of the neighborhoods is relaxed, the constitution of the graph of the image is not constrained, and could for instance be constituted of an arbitrary large number of connected components disconnected the ones from the others. For the problem to be well posed, one would need to work on constraints of topology of the image graph so that the “unreliable” areas would be connected to “reliable” sites.

Finally, the possibility of iterative alternate optimization of the neighborhood and the segmentation has been mentioned, but we did not investigate such an approach further after the observation of oscillatory phenomena.

Appendix A

Synthèse en Français

A.1 Introduction générale

La popularisation des systèmes d'acquisition d'images, de pair avec les avancées technologiques notamment autour des systèmes informatiques et embarqués, a entraîné la multiplication des cadres d'utilisation de l'imagerie numérique, dans de nombreux domaines. Celui de la vision par ordinateur traite du problème de la récupération d'informations haut niveau à partir de données brutes, et avec l'aide de la puissance de calcul des ordinateurs.

Nous pouvons distinguer dans ce domaine de recherche les tâches liées à l'acquisition des données, à leur traitement et à leur analyse. La première catégorie regroupe les techniques liées à la production d'images numériques, par exemple le développement de capteurs pour diverses longueurs d'ondes, diverses échelles, ou avec des propriétés spéciales, tels que les capteurs asynchrones dans le cadre de l'imagerie événementielle. La seconde catégorie s'intéresse aux traitements et modifications des données elles mêmes, afin de les rendre plus facilement exploitables par l'être humain ou par des ordinateurs. Des exemples classiques en sont les algorithmes de débruitage d'images ou de compensation d'exposition.

Finalement, la tâche qui retient notre attention dans le domaine du traitement de l'image est l'analyse des données, qui traite de la décomposition et de la reconnaissance des éléments d'une image. Plus spécifiquement, nous nous intéressons au problème de *segmentation d'image*, dont l'objet est la décomposition d'une image en régions, et l'attribution d'une étiquette à chaque élément. Il s'agit d'une opération complexe, qui relève d'un problème inverse. Un *problème inverse* consiste en l'estimation de la valeur d'une variable aléatoire cachée, ou ici d'un champ aléatoire, à partir d'observations. Il est opposé, en terme de démarche, au *problème direct*, dont le but est de modéliser les observations obtenues à partir d'une variable connue.

Il est typiquement possible par exemple de calculer une segmentation d'image à partir des informations radiométriques contenues dans les éléments d'une image, usuellement des pixels, correspondant chacun à une unité détectable par un capteur. Dans cette thèse néanmoins, nous nous intéressons également au cas où une partition des pixels de l'image, baptisée superpixels, est définie comme décomposition élémentaire de l'image. Lorsque nous souhaitons utiliser une formulation générique compatible avec pixels et superpixels, nous utilisons la notion de *site*.

Dans le cadre général toutefois, la qualité des données mesurées en chacun de ces pixels demeure limitée, d'une part à cause de la faible quantité d'information et d'autre part à cause des altérations pouvant se produire sur ces données, de par des phénomènes optiques ou électroniques, générant un bruit sur la mesure. En intégrant le bruit au modèle, la variable aléatoire cachée ne peut plus être estimée de manière unique, et on se retrouve dans le cadre d'un problème dit *mal posé*.

C'est pourquoi il est usuel de considérer l'acquisition de données comme un tout et de formuler des hypothèses a priori sur la régularité de la variable aléatoire estimée. Pour ce faire, nous nous plaçons dans le cadre spécifique mais très répandu des Champs de Markov (ou Markov Random Fields, ou MRF, [Geman and Geman \[1984\]](#)), dont l'hypothèse principale est l'indépendance

conditionnelle de l'étiquetage de chaque noeud d'un champ par rapport à tous les éléments n'étant pas dans le *voisinage* de ce noeud. A contrario, la probabilité de l'étiquetage de chaque noeud dépend de l'étiquetage de ses voisins: un tel a priori est souvent nommé hypothèse de douceur ou de régularité, mais l'on note aussi que selon les probabilités associées à chaque configuration, cette hypothèse peut aussi permettre de modéliser des solutions ayant tendance à l'hétérogénéité.

Ceci nous amène à évoquer les champs de Gibbs et leur équivalence avec les Champs de Markov, lorsqu'aucune configuration n'est de probabilité nulle (Kindermann and Snell [1980]; Spitzer [1971]). Lorsque cette condition est vérifiée, la probabilité de chaque configuration peut en effet être exprimées en fonction d'une énergie potentielle. Via cette équivalence, et le théorème de Bayes, la maximisation de la probabilité a posteriori d'une segmentation sachant l'observation, correspond à la minimisation d'une énergie F , intégrant dans sa décomposition des termes d'attache aux données E_1 , et un terme de régularisation E_2 , de la forme:

$$F(\mathbf{u}, V) = E_1(\mathbf{u}) + \alpha E_2(\mathbf{u}, V).$$

Si dans la théorie, la régularisation d'une solution telle que décrite vise à s'affranchir du bruit d'acquisition et d'autres phénomènes optiques rendant la segmentation peu fiable, en pratique, elle introduit également des artefacts spécifiques à son implémentation. Le phénomène de régularisation en marche d'escaliers est par exemple une conséquence connue de l'utilisation de la variation totale. L'obtention d'une solution satisfaisante dépend alors souvent d'un compromis, entre le terme d'énergie traduisant l'attache aux données et le terme d'énergie de régularisation, compromis réalisé par l'introduction d'un coefficient dit de régularisation $\alpha \in \mathbb{R}_{>0}$. Les *structures fines* composant une image sont cependant amenées à disparaître bien avant l'élimination de la composante de bruit de l'image, ce qui peut être pénalisant ensuite dans de nombreuses applications.

Une structure fine est une structure dont l'une des dimensions est très faible devant les autres. Par exemple, une structure filiforme de manière générale, telle qu'un crack dans un matériau, le support d'un feu de signalisation sur l'image acquise par un véhicule autonome ou un vaisseau sanguin dans une acquisition d'imagerie médicale en 2 ou 3 dimensions. Mais en dimension 3, les surfaces de faible épaisseur deviennent également des structures fines.

Dans le cadre de l'utilisation de superpixels, qui sont des regroupements de pixels homogènes constituant une partition de l'image, les structures d'une image ont tendance à être constituées d'un faible nombre de superpixels, et deviennent fines relativement à la taille de ces derniers. Or dans de nombreuses applicatins, la préservation de ces structures fines est d'un intérêt crucial. Il devient donc plus délicat de régulariser la composante due au bruit de l'image. D'autres modèles existent, faisant par exemple intervenir dans le terme de régularisation le gradient d'intensité de l'image entre les deux pixels considérés, et visent à remédier à certains des effets négatifs de cette régularisation.

Nous nous penchons quant à nous sur le questionnement et la redéfinition de la notion de voisinage pour son utilisation dans le cadre de la segmentation de structures fines sur des images pouvant être indifféremment constituées de pixels ou de superpixels.

Premièrement, avec l'utilisation de superpixels, la notion de grille régulière de pixels devient caduque, et les éléments de l'image acquièrent des formes et des dispositions irrégulières, ce qui pousse à la redéfinition de ce que peut être la notion de voisinage (voir Figure A.1). Par exemple, Stawiaski and Decencièrre [2011] s'intéressent aux interactions des paires de pixels adjacents au sein de superpixels distincts, et définissent donc un voisinage pondéré par les longueurs de bords communs aux superpixels. À l'inverse, dans un cadre autre que la régularisation, Giraud et al. [2017b] utilise un voisinage isotrope ignorant l'adjacence des superpixels, et défini à partir des positions relatives des barycentres des pixels constituant chaque superpixel.

Secondement, afin de préserver les structures fines, nous concevons des voisinages anisotropes, dont les orientations visent à suivre celles des structures de l'image. Pour ce faire, nous avons implémenté plusieurs méthodes afin de constituer une carte d'orientation et de saillance de ces structures.

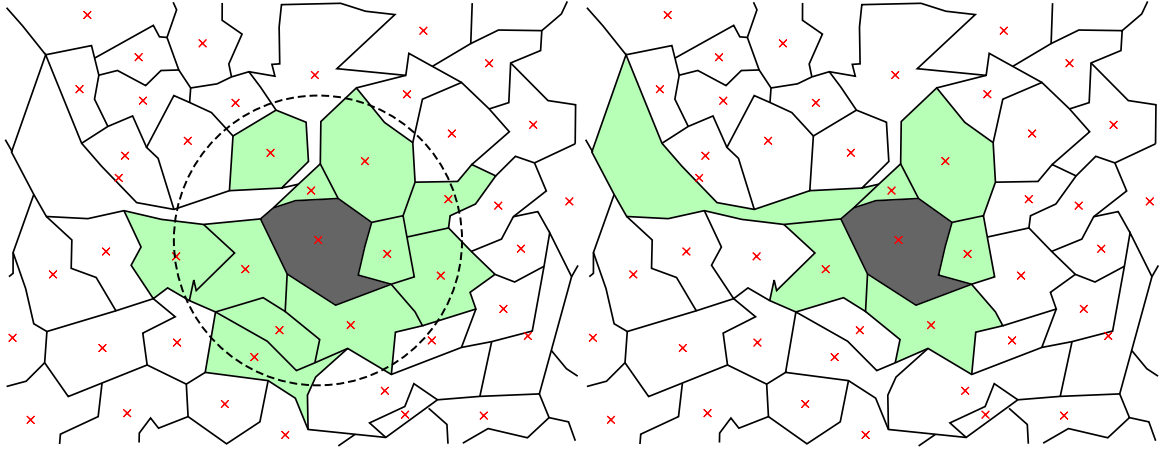


Figure A.1 – Voisinages anisotropes représentés au niveau superpixel. Les voisins du site de référence s (en gris) sont indiqués en vert. Le voisinage à base de disque (à gauche) est défini comme celui qui associe, à chaque site s , l'ensemble des sites dont le barycentre se situe à l'intérieur d'un disque centré sur s . De cette façon, la connexité de $V(s) \cup \{s\}$ n'est pas garantie, ni la forme exacte du voisinage, cependant le positionnement des barycentres l'est. Seules certaines hypothèses sur la convexité et la taille des surfaces des sites permettent d'espérer réduire les configurations atypiques de voisinage. Pour [Stawiaski and Decencière \[2011\]](#) (à droite) en revanche, chaque site adjacent au site de référence est défini comme voisin, ce qui peut entraîner certains sites voisins à s'étendre spatialement arbitrairement loin du site de référence, selon leur forme. Avec cette formulation donc, le voisinage $V(s) \cup \{s\}$ a la garantie d'être constitué d'une unique composante connexe, mais sa forme globale n'est pas maîtrisée.

Troisièmement, nous présentons dans ce manuscrit différentes options pour la construction de ces voisinages anisotropes, cohérents simultanément pour une implémentation niveau pixel ou superpixel.

Enfin, nous mettons à l'épreuve notre approche dans deux applications distinctes, la première étant le cas d'une segmentation binaire sur une image synthétique ainsi que des essais sur la segmentation d'images de cracks. La seconde application concerne le domaine de la reconstitution de cartes de profondeur à partir de séries d'images avec différents niveau de mise au points, aussi appelé *Shape From Focus (SFF)* en anglais.

Dans la partie de ce résumé suivant cette introduction, nous décrirons la construction des voisinages dans un premier temps et décrirons brièvement les expériences et leurs conclusions dans un second temps.

A.2 Voisinages anisotropes

L'obtention de voisinages anisotropes se fait en deux étapes: l'estimation d'une carte d'orientation et de saillance des structures fines tout d'abord, puis la construction des voisinages eux mêmes à partir de cette carte. Pour la première étape, nous avons expérimenté trois méthodes distinctes, qui sont l'approche par minimisation d'une énergie, l'approche par Tensor Voting ([Medioni et al. \[2000\]](#)) et l'approche par RORPO ([Merveille et al. \[2018\]](#)). Concernant la construction du voisinage, nous proposons quatre méthodes, nommées d'après ce sur quoi elles se basent: dans l'ordre de ce manuscrit, il s'agit de l'approche basée sur des formes de voisinages, l'approche basée sur un dictionnaire, et deux approches basées sur des chemins, l'une trouvant le chemin le plus court entre deux sites cibles, et l'autre fixant le cardinal du voisinage. Nous proposons aussi l'utilisation de voisinages isotropes dans les zones homogènes de l'image. Nous détaillons succinctement tout ceci dans cette section.

A.2.1 Estimation de l'orientation

L'estimation de l'orientation des voisinages est une étape cruciale: l'intérêt est de permettre aux voisinages anisotropes de «s'adapter» au contenu de l'image, c'est à dire de suivre l'orientation des structures estimées, et de s'étendre au maximum à l'intérieur de celles ci. C'est seulement à cette condition que l'étape de régularisation pourrait être facilitée, en limitant les phénomènes de dégradation sur le pourtour des structures. Pour réaliser cette étapes, une question importante est le choix des données sur lesquelles se baser. Car en exploitant les mêmes données que celles utilisées pour la segmentation, se pose la question de la convergence de l'approche: si les données d'entrées sont erronées, l'estimation des orientations des voisinages et les voisinages eux mêmes pourrait être dégradée, conduisant à une diminution de la qualité de segmentation. À l'inverse, si l'estimation des structures est robuste aux erreurs, il serait possible de s'appuyer directement sur une segmentation pour en déduire une carte des structures fines, et ainsi améliorer la qualité de la segmentation, ce qui pourrait même être réalisé récursivement, sous réserve de convergence. Une possibilité beaucoup plus complexe que nous n'avons pas explorée serait d'estimer simultanément la segmentation, la carte d'orientation des structures et les voisinages, par une optimisation globale.

Nous nous sommes limités ici à trouver une estimation robuste des orientations à partir d'une segmentation aveugle (non régularisée) et éventuellement des données de l'observation.

La première option d'estimation explorée est celle par *minimisation d'énergie*. Cette approche nécessite d'établir une première segmentation aveugle, et de précalculer un ensemble de voisinages, afin de choisir le voisinage parmi un ensemble de possibilités qui minimise l'hétérogénéité de l'étiquetage de la segmentation dans chaque ensemble de voisins d'un site. En pratique, cela revient à minimiser un terme de Potts, déjà présent dans la fonctionnelle d'énergie F . L'une des limites en revanche serait la sensibilité d'une telle estimation au bruit. Afin de répondre à cette limitation, nous formulons une hypothèse de régularité sur les voisinages eux mêmes, et intégrons un troisième terme d'énergie de régularité des voisinages à la fonctionnelle.

La seconde option explorée fait usage du *vote de tenseurs* (Tensor Voting en anglais, [Medioni et al. \[2000\]](#)), afin de produire une carte d'orientation aux orientations et saillances régulières par construction dans le cas binaire, ou régulière par morceaux dans notre application au Shape From Focus. Chaque tenseur vote dans chaque point de l'espace à sa proximité en suivant une transformation précise, que définit un noyau de vote. Cette transformation consiste en une rotation et une atténuation des valeurs propres du tenseur suivant un comportement assurant la continuité des orientations. Nous nous servons alors des valeurs propres et vecteurs propres de chaque tenseur pour extraire l'orientation et la saillance des structures fines.

La troisième et dernière option avancée est le *RORPO* ([Merveille et al. \[2018\]](#)), qui est un opérateur analysant la tortuosité des structures d'une image à partir d'opérateurs morphologiques tels que l'ouverture de chemins. Cet opérateur réalise sur une image en niveaux de gris un ensemble d'ouvertures de chemins selon différentes orientations, puis compare l'ensemble des images obtenues. La réponse du RORPO est calculée à partir de ces images en discriminant les zones présentant un faible nombre de réponses de fortes intensités: ces zones sont les zones désignées comme présentant des structures anisotropes marquées.

Chacune de ces trois méthode présente des avantages et inconvénients qui lui sont propres, et que nous décrirons dans la dernière section.

A.2.2 Construction du voisinage

Pour la construction à proprement parler du voisinage, nous nous sommes inspirés de deux constructions de voisinages isotropes distinctes au niveau superpixels ([Stawiaski and Decencière \[2011\]](#) et [Giraud et al. \[2017a\]](#)), dont l'une est basée sur les frontières communes entre superpixels (leur adjacence), et l'autre sur les propriétés des barycentres de ceux ci (leur proximité). Nous avons donc repris ces notions pour formuler nos propres voisinages, en rajoutant la prise en compte de la directivité (voir Figure A.2).

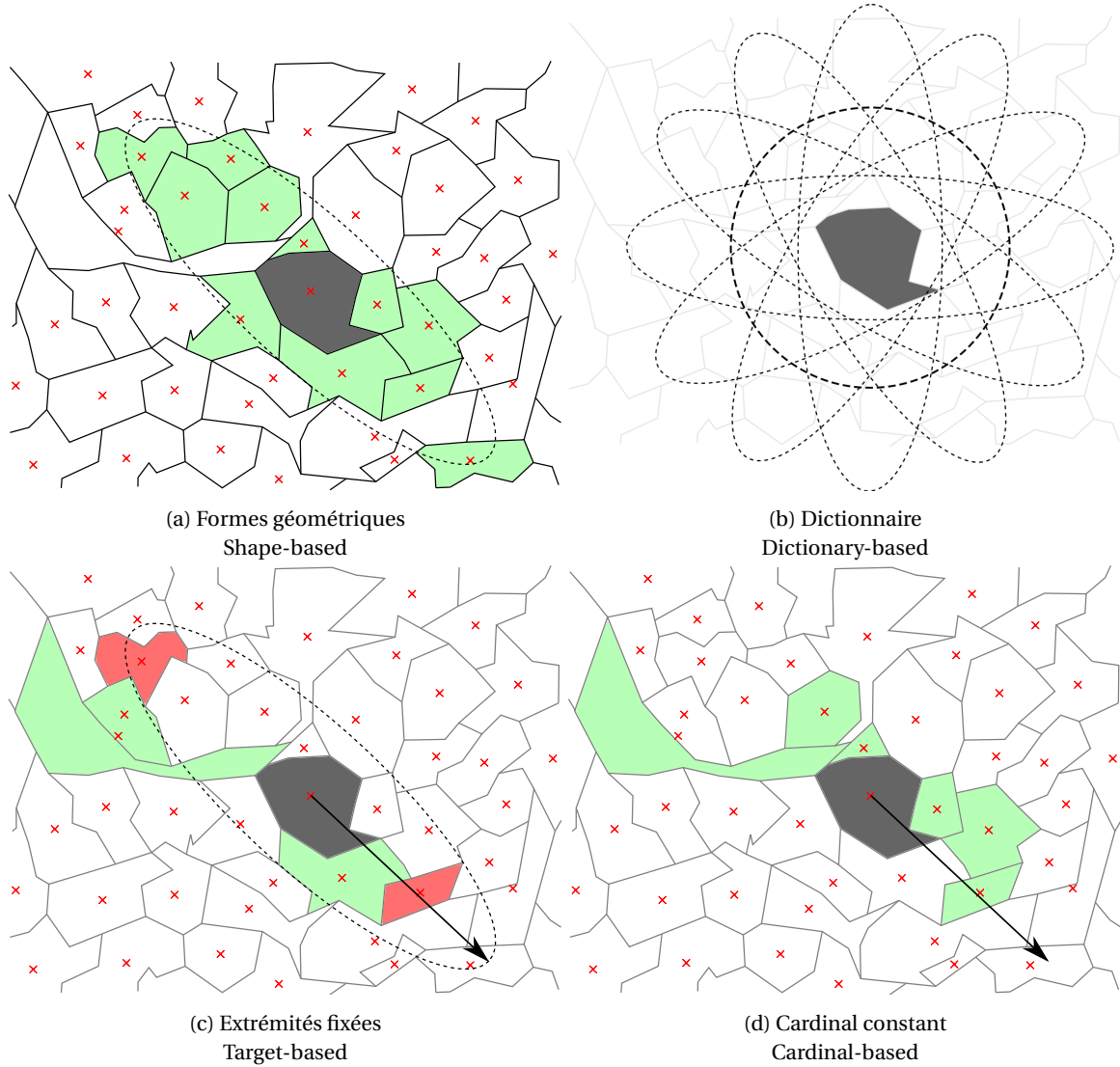


Figure A.2 – Les quatre voisinages anisotropes proposés. La première ligne représente le voisinage basé sur des formes, et le voisinage à base de dictionnaire (illustré sur un dictionnaire de formes). La seconde ligne présente les deux voisinages à base de chemins. À l'exception du voisinage à base de dictionnaires, les voisins $V(s)$ du site s (en gris) sont représentés en vert. Le voisinage à base de **formes** (a) associe à s les sites dont les centroides (les croix rouges) appartiennent à la forme centrée sur s , ici, l'ellipse en pointillés, alignée avec la direction privilégiée. Le voisinage basé sur un **dictionnaire** (b) définit quant à lui un ensemble de configurations possibles avant de déterminer le voisinage $V(s)$ retenu. Ici, dans l'exemple à base de formes, le dictionnaire est constitué d'un ensemble d'ellipses (en pointillés), et le meilleur candidat peut être sélectionné pour chaque site à partir d'une carte d'orientation ou d'une étape d'optimisation (voir notamment Section 4.2). Les voisinages à base de chemins élisent les voisins de chaque site s comme l'union de deux chemins de sites adjacents partant de s , choisis comme les minimiseurs d'une énergie qui tend non seulement à respecter l'orientation définie (représentée par une flèche noire) mais également l'homogénéité radiométrique au sein du chemin. Le voisinage à **extrémités fixées** (c) a pour spécificité de fixer avant tout deux sites (en rouge) comme cibles pour les extrémités des chemins, à l'intérieur de l'ellipse du voisinage à base de formes (en pointillés). Enfin, le voisinage à **cardinal constant** (d) fixe le nombre de voisins à intégrer au voisinage $V(s)$, en définissant une longueur fixe des chemins.

Dans le cas des voisinages basés sur les *formes*, nous considérons des formes géométriques paramétriques (telles que des ellipses) pour définir autour de chaque site une zone où tous les sites possédant un barycentre seraient intégrés au voisinage. Dans le cas d'ellipses par exemple, le demi grand axe de chaque ellipse est aligné avec l'orientation déduite de la carte des orientations et saillances de structures fines.

Le voisinage basé sur un *dictionnaire* est, dans notre implémentation, dérivé d'une discrétisation du voisinage basé sur les formes, mais peut être dans le cas général implémenté pour tout type de voisinage. En pratique, l'idée est de discrétiser les orientations estimées des structures fines, ce qui limite le nombre de configurations de voisinages pour un site donné à un nombre constant et choisi. Cela permet le précalcul des configurations, et a notamment un intérêt dans le cas de l'estimation des structures fines basée sur une énergie découlant de ces configurations.

Les deux dernières options de construction d'un voisinage anisotrope sont basées sur l'adjacence de proche en proche des sites du voisinage, et sont donc des voisinages à base de *chemins*. Le premier voisinage à chemins est construit à partir de *cibles*: pour chaque site, deux sites sont désignés comme voisins à atteindre, et deux chemins les reliant au site source du voisinage sont sélectionnés parmi tous les chemins possibles, en minimisant une énergie spécifique. En particulier, cette énergie fait intervenir les différences radiométriques entre sites mais aussi leur orientation relative, l'idée étant d'encourager l'alignement des sites du chemin mais aussi d'autoriser une flexibilité vis à vis des structures fines. L'union des chemins constitue le voisinage. Le deuxième voisinage à base de chemins, basé sur le *cardinal*, minimise une énergie du même type, mais a pour contrainte la cardinalité du voisinage, au lieu de la connexion de sites distants.

Dans tous les cas, nous permettons aussi aux voisinages de se résoudre sur une formulation isotrope, lorsque la saillance des structures fines est en deçà d'un certain seuil. Les deux voisinages considérés pour cela sont un voisinage basé sur une forme de disque, et le voisinage formulé par [Stawiaski and Decenci re \[2011\]](#).

A.3 Applications et conclusions

Nous nous penchons sur deux cadres applicatifs distincts : le premier est une expérience de segmentation sur un problème binaire, réalisée sur une image synthétique, puis sur une image de cracks, et le deuxième est la reconstitution de cartes de profondeur en SFF.

Au niveau de la segmentation binaire, nous avons mené de nombreuses expériences mêlant l'estimation de structures fines à partir d'une énergie et la construction de voisinages par dictionnaire, avec un grand nombre de formes (ellipses, rectangles, cônes), de tailles, et différentes discrétisations des orientations. En pratique, cette approche s'est révélée présenter des sensibilités aux erreurs d'étiquetage, ainsi qu'être assez imprécise à proximité de l'interface des structures de l'image.

Cela nous a amené à considérer l'approche basée sur le Tensor Voting, pour lequel l'estimation des orientations semble plus probante, avec des voisinages à base de formes, et également à base de chemins. En comparaison chiffrée avec une F-Mesure (mesure basée sur les nombres de faux positifs, faux négatifs, vrai négatifs et vrai positifs), le voisinage à cardinal constant (basé sur des chemins) semble présenter de meilleurs résultats que le voisinage isotrope de [Stawiaski and Decenci re \[2011\]](#) au niveau superpixel pour l'image simulée, les résultats étant toutefois contenus dans un mouchoir de poche au vu du faible nombre d'erreurs (la F-Mesure est comprise en pourcentage dans l'intervalle [98.84, 99.82]).

Bien que demeurant très rapprochés, les résultats dans l'application à la détection de cracks se retrouvent favorables à [Stawiaski and Decenci re \[2011\]](#), et indiquent une certaine sensibilité de l'approche à la qualité des superpixels: en effet, avec les images de cracks, les superpixels ont tendance à être plus irréguliers, et permettent difficilement de segmenter ceux-ci, mais l'utilisation de superpixels basés sur la vérité terrain, nous a permis d'exhiber des situations où nos voisinages anisotropes obtiennent de meilleurs résultats que les voisinages isotropes.

Nous avons alors mené une étude plus complète dans le cadre du SFF, incluant le Tensor Vot-

ing, le RORPO, et les voisinages anisotropes proposés à l'exception de celui à base de dictionnaires. La reconstruction de cartes de profondeur à partir de séries d'images se fait par l'utilisation d'un opérateur de netteté, dont la valeur en un point de l'image est élevée lorsque le contraste est important. En SFF, la profondeur de chaque point de l'image est déduite du maximum de netteté sur la série d'images en ce point.

Dans notre implémentation, nous utilisons le niveau de netteté de la série d'images comme donnée d'entrée pour estimer les cartes d'orientation de structures fines. En définitive, celles produites par le RORPO sont moins régulières que celles produites par Tensor Voting, mais respectent mieux les structures de l'image.

La comparaison des voisinages implémentés semble prouver une supériorité des voisinages isotropes au niveau pixel, tendance qui s'inverse de façon intéressante au niveau superpixel. Finalement, notre comparaison présente aussi les voisinages anisotropes à cardinal constant basés sur le RORPO au niveau superpixel comme étant une des meilleures options pour la régularisation, avec des performances robustes par rapport au paramètre de régularisation tout en ayant des résultats plus performants que les voisinages isotropes. De plus, la comparaison des performances des différentes approches entre les niveaux pixel et superpixel indique que la combinaison du RORPO et des voisinages à cardinal constant au niveau superpixel offre des résultats qui dépassent même ceux obtenus par régularisation isotrope au niveau pixel. Ces résultats sont schématisés dans la Figure 6.36.

Ces résultats sont donc encourageants, et soulèvent des questions qui nous permettent d'envisager des perspectives intéressantes pour de futurs travaux. En effet, de nombreux points possiblement soumis à étude n'ont pas pu être abordés, suite à notre priorisation sur le sujet. Notamment, nous pourrions évoquer l'impact du choix et du nombre de superpixels, et de leurs pré-traitements. La question de la régularisation des cartes de profondeur pourrait aussi être posée, notamment dans le cas du RORPO, fidèle sur les pourtours des objets, mais légèrement bruité en leur centre et lorsque la netteté est faible. L'ensemble des paramètres visant à construire la carte des structures fines, dont la vérité terrain est inconnue, tels que la longueur de l'ouverture des chemins, ou les paramètres du Tensor Voting, a été fixée empiriquement après des observations qualitatives, mais une étude approfondie pourrait être menée. Au niveau de la construction des voisinages, les mêmes questions se posent, le nombre de paramètres jouant sur la qualité des voisinages étant relativement important. Par ailleurs, une question intéressante se pose vis à vis des voisinages et de leurs cliques, par rapport à la constitution finale du graphe constituant l'image. En effet, le graphe de l'image n'est pas contraint, par exemple, à ne constituer qu'une composante connexe, et des nombres arbitraires de noeuds (ou sites), peuvent se retrouver isolés des autres, par paquets de taille tout aussi arbitraire. Il pourrait être intéressant par exemple, lorsque l'on sait que certaines zones sont assujetties au bruit, de s'affranchir de l'effet terme d'attache aux données et s'introduire une contrainte de construction des voisinages telle que ces zones soient connectées à des zones «fiabiles». Ceci permettrait d'assurer la convergence de telles zones vers des valeurs cohérentes. Enfin, nous avons évoqué la possibilité d'une optimisation itérative du voisinage, que nous n'avons pas étudié en détail après la constatation de phénomènes oscillatoires.

Appendix B

Demonstrations

B.1 Energy based guidance map

In this section we prove the fact that the energy E_3^{st} defined in Equation (3.8) (replicated below) is a pseudometric. Since it is a pseudometric, the triangle inequality yields the submodularity required for using α -expansion or α - β swap.

B.1.1 Pseudometric

$$E_3^{st}(\mathbf{g}_s, \mathbf{g}_t) = \begin{cases} \arccos(|\leq \mathbf{g}_s, \mathbf{g}_t \geq|) & \text{if } \mathbf{g}_s \neq 0 \text{ and } \mathbf{g}_t \neq 0, \\ \arcsin(|\leq \mathbf{g}_s, \vec{st} \geq|) & \text{if } \mathbf{g}_s \neq 0 \text{ and } \mathbf{g}_t = 0, \\ \arcsin(|\leq \vec{st}, \mathbf{g}_t \geq|) & \text{if } \mathbf{g}_s = 0 \text{ and } \mathbf{g}_t \neq 0, \\ 0 & \text{if } \mathbf{g}_s = 0 \text{ and } \mathbf{g}_t = 0. \end{cases} \quad (\text{B.1})$$

Let $a, b, c \in \mathbb{R}^2$ be three vectors of \mathbb{R}^2 . For proving that E_3^{st} is a pseudometric, we need to prove three preperities:

$$\begin{aligned} E_3^{st}(a, a) &= 0, \\ E_3^{st}(a, b) &= E_3^{st}(b, a), \\ E_3^{st}(a, c) &\leq E_3^{st}(a, b) + E_3^{st}(b, c). \end{aligned}$$

The first property is trivially verified, since whatever the value of $a \in \mathbb{R}^2$, either $E_3^{st}(a, a) = \arccos(1) = 0$ or $E_3^{st}(\vec{0}, \vec{0}) = 0$. The verification of the second property is also trivial, since $\leq a, b \geq = \leq b, a \geq$.

Now, we need to check whether E_3^{st} verifies triangular inequality. We split the cases depending if some vectors are null or not. When $a = b = c = \vec{0}$, when $a = b = \vec{0}$ or when $b = c = \vec{0}$, the triangular inequality is verified since $E_3^{st}(a, c) = E_3^{st}(a, b) + E_3^{st}(b, c)$ in those cases, thanks to the first property. When $a = c = \vec{0}$, the triangular inequality is also verified since the energy is non negative and $E_3^{st}(a, c) = 0$. This leaves four configurations to explore.

First configuration If neither of the vectors is null, we consider $a, b, c \in \mathbb{R}^2 \setminus \{\vec{0}\}$, and define the following oriented angles $x, y, z \in \mathbb{R}$, constructing them wisely such that:

$$\begin{cases} y = (\widehat{a, b}) & \in [0, \pi], \\ x = y + z & \in [-\frac{\pi}{2}, \frac{3\pi}{2}], \\ z = (\widehat{b, c}) & \in [-\frac{3\pi}{2}, \frac{3\pi}{2}], \end{cases}$$

where $(\widehat{a, b})$ is the angle between the vectors a and b . We note that $x \equiv (\widehat{a, c}) \pmod{2\pi}$.

Then, for convenience, we write:

$$\begin{aligned} A &= \arccos(|\leq a, c \geq|), \\ B &= \arccos(|\leq a, b \geq|), \\ C &= \arccos(|\leq b, c \geq|). \end{aligned}$$

We note that, $\forall a, b \in \mathbb{R}^2, \leq a, b \geq = \cos(\widehat{a, b})$, and $\forall t \in \mathbb{R}$,

$$\arccos(|\cos(t)|) = \left| t - \left\lfloor \frac{t}{\pi} + \frac{1}{2} \right\rfloor \pi \right|,$$

therefore,

$$\begin{cases} A = |x|, & \text{if } |x| < \frac{\pi}{2}, \\ A = |\pi - x|, & \text{if } x \in [\frac{\pi}{2}, \frac{3\pi}{2}] \end{cases}, \quad \begin{cases} B = y, & \text{if } y \in [0, \frac{\pi}{2}], \\ B = \pi - y, & \text{if } y \in [\frac{\pi}{2}, \pi] \end{cases}, \quad \begin{cases} C = |z|, & \text{if } |z| < \frac{\pi}{2}, \\ C = |\pi - z|, & \text{if } z \in [\frac{\pi}{2}, \frac{3\pi}{2}], \\ C = |\pi + z|, & \text{if } z \in [-\frac{3\pi}{2}, -\frac{\pi}{2}]. \end{cases}$$

The verification of the triangular inequality is then made by checking that $A \leq B + C$ for any combination of those values, which yields 9 cases.

- If $0 \leq y \leq \frac{\pi}{2}$, and $|x| \leq \frac{\pi}{2}$, since $x = y + z$, we know that $\frac{\pi}{2} \geq \frac{\pi}{2} - y \geq z \geq -\frac{\pi}{2} - y \geq -\pi$.

- If $|z| \leq \frac{\pi}{2}$, we have the triangular inequality:

$$B + C = |y| + |z| \geq |y + z| = |x| = A.$$

- If $-\frac{\pi}{2} \geq z \geq -\frac{\pi}{2} - y$, then

$$B + C = y + \pi + z \geq y + \pi - \frac{\pi}{2} - y = \pi/2 > |x| = A.$$

- If $0 \leq y \leq \frac{\pi}{2}$, and $\frac{\pi}{2} \leq x \leq 3\frac{\pi}{2}$, we know that $3\frac{\pi}{2} \geq x \geq z \geq x - \frac{\pi}{2} \geq 0$.

- If $\frac{\pi}{2} \leq z \leq 0$ since $x \geq \frac{\pi}{2}$, $x \geq |\pi - x|$ and:

$$B + C = y + z = x \geq |\pi - x| = A.$$

- If $3\frac{\pi}{2} \geq x \geq z \geq \frac{\pi}{2}$:

$$B + C = y + |z - \pi| \geq |y + z - \pi| = |x - \pi| = A.$$

- If $\pi \geq y \geq \frac{\pi}{2}$, and $|x| \leq \frac{\pi}{2}$, we know that $0 \geq \frac{\pi}{2} - y \geq z \geq -\frac{\pi}{2} - y \geq -3\frac{\pi}{2}$.

- If $\frac{\pi}{2} - y \geq z \geq -\frac{\pi}{2}$,

$$B + C = \pi - y - z \geq \pi - y + y - \frac{\pi}{2} = \frac{\pi}{2} \geq |x| = A.$$

- If $-\frac{\pi}{2} \geq z \geq -3\frac{\pi}{2}$:

$$B + C = \pi - y + |\pi + z| \geq |\pi - y - \pi - z| = |-x| = A.$$

- If $\pi \geq y \geq \frac{\pi}{2}$ and $3\frac{\pi}{2} \geq x \geq \frac{\pi}{2}$, we know that $\pi \geq 3\frac{\pi}{2} - y \geq z \geq \frac{\pi}{2} - y \geq -\frac{\pi}{2}$.

- If $\frac{\pi}{2} \geq z \geq \frac{\pi}{2} - y$,

$$B + C = \pi - y + |z| \geq |\pi - y - z| = |\pi - x| = A.$$

- If $3\frac{\pi}{2} - y \geq z \geq \frac{\pi}{2}$, we have two cases to distinguish:

- * if $\pi \geq x \geq \frac{\pi}{2}$,

$$B + C = \pi - y + |\pi - z| \geq |2\pi - x| = 2\pi - x \geq \pi - x = A.$$

- * if $3\frac{\pi}{2} \geq x \geq \pi$

$$B + C - A = \pi - y + |\pi - z| - (x - \pi) \geq |2\pi - x| + \pi - x = 3\pi - 2x.$$

$$\text{However, } 2x \leq 3\pi, \text{ therefore } B + C - A \geq 0, \text{ and } B + C \geq A.$$

This proves the triangular inequality when $a, b, c \in \mathbb{R}^2 \setminus \{\vec{0}\}$.

Second configuration Now, when $b = \vec{0}$ and $a, c \in \mathbb{R}^2 \setminus \{\vec{0}\}$, note that $\forall t \in \mathbb{R}$,

$$\arcsin(|\cos(t)|) = \arcsin\left(\left|\sin\left(t + \frac{\pi}{2}\right)\right|\right) = \arcsin\left(\left|\sin\left(t - \frac{\pi}{2}\right)\right|\right) = \arccos\left(\left|\cos\left(t - \frac{\pi}{2}\right)\right|\right),$$

and we suppose that $\vec{st} \neq \vec{0}$. Let us define $b' \in \mathbb{R}^2 \setminus \{\vec{0}\}$ such that $\langle \vec{st}, b' \rangle = 0$. Based on the foregoing,

$$\arccos(|\leq a, c \geq|) \leq \arccos(|\leq a, b' \geq|) + \arccos(|\leq b', c \geq|), \quad (\text{B.2})$$

$\forall a, b', c \in \mathbb{R} \setminus \{\vec{0}\}$. Similarly to the previous development, we define the following oriented angles $x, y', z' \in \mathbb{R}$, constructing them wisely such that:

$$\begin{cases} y' = \widehat{(a, b')} & \in [0, \pi], \\ x = y' + z' & \in [-\frac{\pi}{2}, \frac{3\pi}{2}], \\ z' = \widehat{(b', c)} & \in [-\frac{3\pi}{2}, \frac{3\pi}{2}], \end{cases}$$

and note that $x \equiv \widehat{(a, c)} \pmod{2\pi}$. Equation (B.2) becomes:

$$\arccos(|\leq a, c \geq|) \leq \arccos(|\cos(y')|) + \arccos(|\cos(z')|). \quad (\text{B.3})$$

Since the angles y' and z' are oriented angles, and \vec{st} and b' are orthogonal, we now consider the following angles:

$$\begin{cases} y = \widehat{(a, \vec{st})} = y' \pm \frac{\pi}{2} \\ z = \widehat{(\vec{st}, c)} = z' \mp \frac{\pi}{2}. \end{cases}$$

Finally, we obtain the triangular inequality from the definition of the energy Equation (B.1) together with Equation (B.3),

$$\begin{aligned} E_3^{st}(a, b) + E_3^{st}(b, c) &= \arcsin(|\leq a, \vec{st} \geq|) + \arcsin(|\leq \vec{st}, c \geq|) \\ &= \arcsin(|\cos(y)|) + \arcsin(|\cos(z)|) \\ &= \arccos(|\cos(y')|) + \arccos(|\cos(z')|) \\ &\geq \arccos(|\leq a, c \geq|) \\ &\geq E_3^{st}(a, c). \end{aligned}$$

Third and fourth configurations In those configurations, only one vector out of a and c is null. Since the energy is symmetrical, $E_3^{st}(a, c) = E_3^{st}(c, a)$ and it is trivial that if one configuration verifies the triangular inequality, the second does it as well. In the latter, we suppose $a, b \in \mathbb{R}^2 \setminus \{\vec{0}\}$ and $c = \vec{0}$. The demonstration is very similar to the previous configuration, since we define a vector $c' \in \mathbb{R}^2 \setminus \{\vec{0}\}$ such that $\langle \vec{st}, c' \rangle = 0$.

We define the following oriented angles $x', y, z' \in \mathbb{R}$, constructing them wisely such that:

$$\begin{cases} y = \widehat{(a, b)} & \in [0, \pi], \\ x' = y + z' & \in [-\frac{\pi}{2}, \frac{3\pi}{2}], \\ z' = \widehat{(b, c')} & \in [-\frac{3\pi}{2}, \frac{3\pi}{2}], \\ z = z' \pm \frac{\pi}{2}, \text{ and} \\ x = x' \pm \frac{\pi}{2}. \end{cases}$$

Given the above, it is straightforward that

$$\begin{aligned} \arccos(|\cos(x')|) &\leq \arccos(|\cos(y)|) + \arccos(|\cos(z')|) \\ \Rightarrow \arcsin(|\sin(x')|) &\leq \arccos(|\cos(y)|) + \arcsin(|\sin(z')|) \\ \Rightarrow \arcsin(|\cos(x)|) &\leq \arccos(|\cos(y)|) + \arcsin(|\cos(z)|) \\ \Rightarrow E_3^{st}(a, c) &\leq E_3^{st}(a, b) + E_3^{st}(b, c). \end{aligned}$$

We have proven in this section of the appendix that the energy E_3^{st} in Section 3.2.1 and Equation (3.8) is a pseudo-metric, and therefore all the subproblems involved in the minimization of this energy are submodular, which allows for using graph cuts as presented in Section 4.1.3.

B.2 Tensor Voting

B.2.1 Geometry and stick kernel

The aim of the calculus of this section is to express the decay function $DF(\hat{\mathbf{e}}_0, \overrightarrow{OP})$ and the rotation vector $\mathbf{\Omega}(\hat{\mathbf{e}}_0, \overrightarrow{OP})$, and finally to compute the rotation matrix $\mathbf{R}(\mathbf{\Omega})$.

We recall that $\hat{\mathbf{e}}_0 \in \mathbb{R}^3$ is the eigenvector of the stick tensor $\mathbb{T}_{stick} = \hat{\mathbf{e}}_0 \hat{\mathbf{e}}_0^T$, and $\overrightarrow{OP} \in \mathbb{R}^3$ is the displacement between the tensor that votes in O, and the tensor or site where is cast the vote located at the point P. The point O and the vectors $\hat{\mathbf{e}}_0$ and \overrightarrow{OP} define altogether the plane of Figure B.1, figure that has already been introduced in Section 3.2.2.

Decay function and rotation vector computation

The definitions given by Medioni et al. [2000] mention the decay function as follows (see Equation (3.10)):

$$DF(r, \phi, \sigma_T) = \exp\left(-\frac{r^2 + \nu\phi^2}{\sigma_T^2}\right),$$

where r is defined as the curvilinear abscissa of the osculating circle passing by O and P, with normal $\hat{\mathbf{e}}_0$ in O, and ϕ is defined as the angle between $\hat{\mathbf{e}}_0$ and $\hat{\mathbf{e}}'_0$. The latter vector is the eigenvector with non null eigenvalue of the tensor cast at location P. Given that $\hat{\mathbf{e}}'_0$ is also normal to the osculating circle, in point P, this also gives us an expression of $\mathbf{\Omega}$:

$$\mathbf{\Omega} = \arccos(\langle \hat{\mathbf{e}}_0, \hat{\mathbf{e}}'_0 \rangle) \frac{\hat{\mathbf{e}}_0 \times \hat{\mathbf{e}}'_0}{\|\hat{\mathbf{e}}_0 \times \hat{\mathbf{e}}'_0\|}, \text{ when } \hat{\mathbf{e}}'_0 \neq \pm \hat{\mathbf{e}}_0.$$

The explication of such expression is trivial. Since the rotation vector transforms the vector $\hat{\mathbf{e}}_0$ into $\hat{\mathbf{e}}'_0$, it is necessarily orthogonal to both vectors if they are non collinear, so $\mathbf{\Omega}$ is collinear to their vector product. Since $(\hat{\mathbf{e}}_0, \hat{\mathbf{e}}'_0, \hat{\mathbf{e}}_0 \times \hat{\mathbf{e}}'_0)$ is a direct basis by construction, the angle of rotation, which is the norm of $(\mathbf{\Omega})$, is comprised in $[0, \pi]$. Since it is also the angle between $\hat{\mathbf{e}}_0$ and $\hat{\mathbf{e}}'_0$, it is effectively computed by the arc cosinus of their scalar product.

In the specific case where $\hat{\mathbf{e}}'_0 = \hat{\mathbf{e}}_0$, there is no rotation and $\mathbf{\Omega} = \vec{0}$. In the case $\hat{\mathbf{e}}'_0 = -\hat{\mathbf{e}}_0$, and any rotation vector orthogonal to $\hat{\mathbf{e}}_0$ with norm π may be used. However this case is not supposed to happen since the voting kernels are restricted to the space where $\|\mathbf{\Omega}\| < \pi/2$.

Now, we note that $\hat{\mathbf{e}}'_0$ is not given and must be deduced from other data, as well as r and ϕ in the expression of DF. The deduction is purely geometric from the elements presented in Figure B.1:

- Since C is the center of the osculating circle passing through O (normal $\hat{\mathbf{e}}_0$) and P: $OC = CP$, C belongs to the mediator of $[OP]$ that is also the bisector of \widehat{OCP} .
- A, B are set on this mediator, such that B is the middle of $[OP]$ and COA is a rectangle triangle.
- We note that (AC) is a symmetry axis for the elements of the figure.
- At point A, we note that $\phi = \pi - 2\widehat{OAB}$.
- In the rectangle triangle OBA, $\widehat{OAB} = \pi/2 - \widehat{BOA}$.
- At point O, we can compute $\widehat{BOA} = \pi/2 - \arccos(\langle \hat{\mathbf{e}}_0, \overrightarrow{OP} \rangle)$.
- Thus, $\boxed{\phi = \pi - 2 \arccos(\langle \hat{\mathbf{e}}_0, \overrightarrow{OP} \rangle)}$
- The arc of circle's length is derived from the perimeter of the circle:
 $r = \frac{2\widehat{OCB}}{2\pi} OC$.
- In the rectangle triangle OCB, $\widehat{OCB} = \pi/2 - \arccos(\langle \hat{\mathbf{e}}_0, \overrightarrow{OP} \rangle) = \widehat{BOA}$.

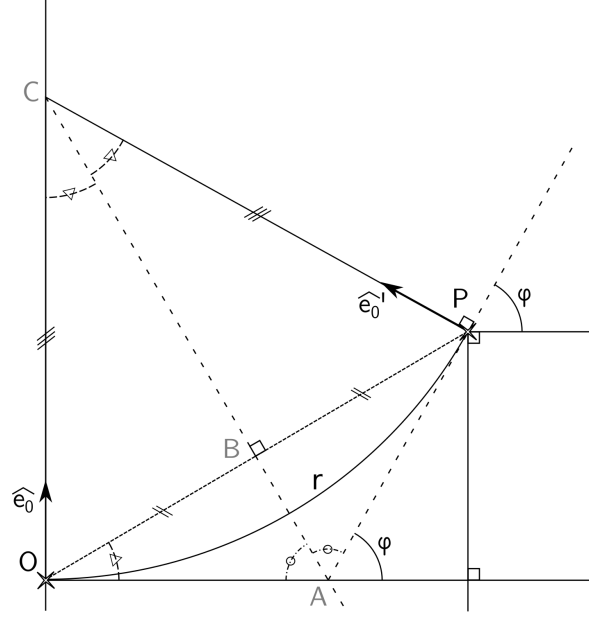


Figure B.1 – Notations for stick voting kernel. The tensor localized in O with normal $\hat{\mathbf{e}}_0$ casts its vote at point P . The result is a stick tensor which normal is $\hat{\mathbf{e}}'_0$, with its eigenvalue scaled by the decay function DF . This decay function is computed from the length r of the arc of the osculating circle passing from O to P with normal $\hat{\mathbf{e}}_0$, and from the angle ϕ . We note three important points, A , B , C , used for constructing the figure and writing explicit angles. Here, (AC) is the mediator of $[OP]$ and crosses this segment in B . All of these geometrical values can be computed from \overrightarrow{OP} and $\hat{\mathbf{e}}_0$.

- In the same triangle, from the cosine of \widehat{COB} : $OC = \frac{OP/2}{|\langle \hat{\mathbf{e}}_0, \overrightarrow{OP} \rangle|}$.
- Therefore,
$$r = \frac{\pi - 2 \arccos(\langle \hat{\mathbf{e}}_0, \overrightarrow{OP} \rangle)}{4\pi |\langle \hat{\mathbf{e}}_0, \overrightarrow{OP} \rangle|} \|\overrightarrow{OP}\|.$$

This allows for computing $DF(\hat{\mathbf{e}}_0, \overrightarrow{OP})$ from the definition of [Medioni et al. \[2000\]](#). However, we still need to compute $\hat{\mathbf{e}}'_0$, and we do so by considering the symmetry with $\hat{\mathbf{e}}_0$ with respect to (AC) :

- By symmetry, $\langle \hat{\mathbf{e}}_0, \overrightarrow{AC} \rangle = \langle \hat{\mathbf{e}}'_0, \overrightarrow{AC} \rangle$, so $\langle \hat{\mathbf{e}}_0 - \hat{\mathbf{e}}'_0, \overrightarrow{AC} \rangle = 0$.
- Since (OP) is orthogonal to (AC) , we can decompose $\hat{\mathbf{e}}_0$ and $\hat{\mathbf{e}}'_0$ by their projections:

$$\hat{\mathbf{e}}_0 - \hat{\mathbf{e}}'_0 = \langle \hat{\mathbf{e}}_0 - \hat{\mathbf{e}}'_0, \overrightarrow{AC} \rangle \frac{\overrightarrow{AC}}{\|\overrightarrow{AC}\|^2} + \langle \hat{\mathbf{e}}_0 - \hat{\mathbf{e}}'_0, \overrightarrow{OP} \rangle \frac{\overrightarrow{OP}}{\|\overrightarrow{OP}\|^2}$$

- Therefore,
$$\hat{\mathbf{e}}'_0 = \hat{\mathbf{e}}_0 - 2 \langle \hat{\mathbf{e}}_0, \overrightarrow{OP} \rangle \frac{\overrightarrow{OP}}{\|\overrightarrow{OP}\|^2}.$$

Now we can compute $\mathbf{\Omega}$ from $\hat{\mathbf{e}}_0$ and \overrightarrow{OP} .

Rotation matrix and rotation of tensors

In this section we present the different ways we compute the rotations of vectors, matrices and tensors, in that manuscript.

When the rotation is defined by *an axis and an angle* (represented by a rotation vector $\mathbf{\Omega} \neq \vec{0}$), the formula we use is the following one. Let ω be the norm of the vector (and therefore the angle of rotation), $\omega = \|\mathbf{\Omega}\| \in \mathbb{R}_{>0}$, and let $v_x, v_y, v_z \in \mathbb{R}^3$ be the components of $\frac{\mathbf{\Omega}}{\omega}$. We write the rotation

matrix as follows:

$$\mathbf{R}(\boldsymbol{\Omega}) = \begin{pmatrix} \cos(\omega) + \nu_x^2(1 - \cos(\omega)) & \nu_x \nu_y(1 - \cos(\omega) - \nu_z \sin(\omega)) & \nu_x \nu_z(1 - \cos(\omega) + \nu_y \sin(\omega)) \\ \nu_x \nu_y(1 - \cos(\omega) + \nu_z \sin(\omega)) & \cos(\omega) + \nu_y^2(1 - \cos(\omega)) & \nu_y \nu_z(1 - \cos(\omega) - \nu_x \sin(\omega)) \\ \nu_x \nu_z(1 - \cos(\omega) - \nu_y \sin(\omega)) & \nu_y \nu_z(1 - \cos(\omega) + \nu_x \sin(\omega)) & \cos(\omega) + \nu_z^2(1 - \cos(\omega)) \end{pmatrix}. \quad (\text{B.4})$$

When $\boldsymbol{\Omega} = \vec{0}$, we note that this reduces to the identity matrix.

We notice that

$$\begin{aligned} \det(\mathbf{R}(\boldsymbol{\Omega})) &= 1 \\ \mathbf{R}(\boldsymbol{\Omega})\mathbf{R}(\boldsymbol{\Omega})^T &= \mathbf{I}_3 \\ \mathbf{R}(-\boldsymbol{\Omega}) &= \mathbf{R}(\boldsymbol{\Omega})^T = \mathbf{R}(\boldsymbol{\Omega})^{-1}, \end{aligned}$$

where \cdot^T is the transposition operation and \mathbf{I}_3 is the identity matrix in $\mathbb{R}^{3 \times 3}$. With this definition, $\mathbf{R}(\boldsymbol{\Omega})$ is a rotation matrix and belongs to the orthogonal group in dimension 3.

The formula for rotating a vector $\mathbf{v} \in \mathbb{R}^3$ with such matrix is $\boxed{\mathbf{v}' = \mathbf{R}\mathbf{v}}$.

However, when it comes to a tensor, and the matrix $\mathbb{M} \in \mathbb{R}^{3 \times 3}$ that represents it, multiple options coexist when it comes to rotations or change-of-base formulas. Depending of the formulation, the elements of the tensor may be covariant, contravariant, or both, and this leads, in the general case, to different formulas for rotating them.

For instance, when considering that a matrix \mathbb{M} embeds an endomorphism $M : \mathbb{R}^3 \rightarrow \mathbb{R}^3$, it maps a vector space to itself, and therefore we can write, $\forall \mathbf{v} \in \mathbb{R}^3, \exists \mathbf{w} \in \mathbb{R}^3$ such that $\mathbb{M}\mathbf{v} = \mathbf{w}$. Since we consider a rotation of such endomorphism, we now consider the new equation:

$$\mathbb{M}'(\mathbf{R}\mathbf{v}) = \mathbf{R}(\mathbb{M}\mathbf{v}).$$

Since it is true $\forall \mathbf{v} \in \mathbb{R}^3$, we can write:

$$\mathbb{M}'\mathbf{R} = \mathbf{R}\mathbb{M},$$

and, if \mathbf{R} is an invertible matrix, we have:

$$\mathbb{M}' = \mathbf{R}\mathbb{M}\mathbf{R}^{-1}.$$

Now, if the matrix \mathbb{M} embeds a bilinear form $M : (\mathbb{R}^3 \times \mathbb{R}^3) \rightarrow \mathbb{R}$, it means that for any pair of vectors $(\mathbf{v}, \mathbf{w}) \in \mathbb{R}^{3^2}$, the value $M(\mathbf{v}, \mathbf{w}) = \mathbf{v}^T \mathbb{M} \mathbf{w}$ is fixed. When we consider the rotation of these vectors by change of base, we have:

$$\begin{aligned} (\mathbf{R}\mathbf{v})^T \mathbb{M}'(\mathbf{R}\mathbf{w}) &= \mathbf{v}^T \mathbb{M} \mathbf{w}, \text{ and so} \\ \mathbf{v}^T (\mathbf{R}^T \mathbb{M}' \mathbf{R}) \mathbf{w} &= \mathbf{v}^T \mathbb{M} \mathbf{w}. \end{aligned}$$

Since this is true $\forall (\mathbf{v}, \mathbf{w}) \in \mathbb{R}^{3^2}$, we can write:

$$\mathbb{M}' = (\mathbf{R}^T)^{-1} \mathbb{M} \mathbf{R}^{-1},$$

if and only if \mathbf{R} and \mathbf{R}^T are invertible.

We can see that the two expressions are, in the general case, non equivalent. However, since in the orthogonal group, to which belongs $\mathbf{R}(\boldsymbol{\Omega})$ in our case, the transpose of a matrix is also its inverse, we have the expression of the rotation used in tensor voting $\boxed{\mathbb{M}' = \mathbf{R}\mathbb{M}\mathbf{R}^T}$.

Appendix C

Submitted article

C.1 Introduction

For many image processing problems such as image segmentation or reconstruction, low-level information delivered by a single pixel is limited and prone to noise, corrupted data and all kinds of optic phenomena altering the original image. Therefore, taking into account a statistical relationship between spatially close pixels has been introduced relatively early in image processing [Geman and Geman \[1984\]](#). A classical way to handle this is to model the two-dimensional (2D) field of pixels as a Markov Random Field (MRF). This allows for introducing a prior on the expected solution. Variational approaches are particularly used to provide solutions, by combining the prior and conditional probabilistic models into a single parametric functional to be minimized. However, due to the dimensionality of the solution space and depending on the form of the functional, finding a global minimizer of it often appears as a challenging task. The study [Szeliski et al. \[2008\]](#) gives an insight by comparing several minimization algorithms (including graph cuts) on typical vision problems (including image segmentation and image reconstruction). It is well established and documented that standard Total Variation (TV) regularization (e.g. in image reconstruction [Ribal et al. \[2018\]](#)) or Potts regularization (e.g. in image segmentation [Boykov and Jolly \[2001\]](#)) using isotropic neighborhoods behave poorly on thin structures. Although they are ubiquitous in a number of applications, their detection remains very difficult because of their spatial sparsity, their small size and their potential complex geometry. Since these structures essentially consist of discontinuities, standard TV and Potts regularization tend to early remove them as regularization increases and are thus not adapted to handle them correctly [Favaro \[2010\]](#).

In parallel with algorithmic developments, the

volume and the diversity of data to exploit have greatly increased over the last years, therefore pre-processing have been proposed to reduce the computational burden. For instance, superpixel decomposition methods [Stutz et al. \[2018\]](#) have been developed for grouping pixels sharing similar radiometric intensities into homogeneous regions, and then drastically reducing the number of elements to process while preserving the geometrical information that is lost with multi-resolution approaches. For instance and specifically for segmentation problem, [Arbeláez et al. \[2011\]](#); [Gould et al. \[2009\]](#) grow and merge regions from an initial set of superpixels that they call an over-segmentation of the initial image. A major drawback of a superpixel segmentation is that the usual hypothesis of a regular topological lattice is lost, as well as the regularity in size and shape of every lattice element. As a result, image segmentation approaches taking advantage of superpixels must cope with these problems and introduce new methods and spatial relationships. Often, superpixels are considered as neighbors when sharing a common border [Cui et al. \[2018\]](#); [Fulkerson et al. \[2009\]](#); [Liu et al. \[2016\]](#); [Stawiaski and Decencière \[2011\]](#). The authors of [Stawiaski and Decencière \[2011\]](#) propose to minimize an energy via graph cuts based on the adjacency graph obtained from the watershed segmentation, where edges connecting two regions are weighted upon the common border length between these regions. Similarly, [Cui et al. \[2018\]](#) propose to ease the classification of the high-dimensional noisy hyperspectral images by building a weighted graph based on superpixels. In [Pei et al. \[2014\]](#), the authors compute saliency from MRF using the same concept of adjacency, and take into account in their algorithm the second-order neighborhood to ease the propagation of information between superpixels. Other superpixel approaches use patches to analyze the spatial con-

tent over a neighboring window and find the nearest matches in a set of reference patches [Giraud et al. \[2017a\]](#). In [Yu et al. \[2015\]](#), the authors train a deep Hough forest from a set of superpixel patches in order to detect objects in aerial images.

Although our work on anisotropic neighborhood can be applied to several segmentation or reconstruction problems, we focus on the application Shape From Focus (SFF) in this paper. SFF is a popular method used for inferring the 3D shape of an object from a set of images with varying focus settings [Nayar and Nakagawa \[1994\]](#). Such an approach only requires one fixed camera with a rather short depth of field and is able to move this camera or to change the focal distance of the optical system. SFF is therefore applicable in many real world applications including industrial inspection, micro manufacturing, robotic control, 3D model reconstruction, medical imaging systems and microscopy. In addition to its intrinsic interest, we focus on this application to illustrate the benefit of anisotropic neighborhoods since naive pixel-level estimates are hampered by the presence of homogeneous surfaces, thus requiring some regularization to propagate the information from reliable areas to uncertain ones, while preserving thin structures. However, the regularization is all the more challenging that the number of labels (i.e., the number of discrete depth values) is important and that structures (and input data) are 3D.

Our first contribution is to propose different estimations of anisotropic neighborhoods on an irregular lattice such as the ones provided by superpixel segmentation. Our second contribution is to propose SFF based on superpixels. It allows us to illustrate the benefit of anisotropic neighborhood since SFF represents a sufficiently complex application so that results may depend on the considered neighborhood.

The rest of this paper is organized as follows. In Section C.2, we specify the considered problem, namely SFF using superpixel segmentation. In Section C.3, we detail the proposed path-based constructions of anisotropic neighborhoods, based on a preliminary estimation of local anisotropy and orientation either from Tensor Voting [Medioni et al. \[2000\]](#), or from RORPO [Merveille et al. \[2018\]](#). Section C.4 discusses the results and benefits of our approach in a comprehensive comparative study between isotropic and anisotropic neighborhoods both in terms of accuracy and time complexity. Finally, Section C.5 draws main conclusions and perspectives.

C.2 Superpixels-based SFF

C.2.1 Basics of SFF

The core idea of SFF is that the closer an object is to the object focal plane (i.e., the more it is focused), the more it appears sharp. Conversely, the farther an object is from this object focal plane, the more it appears blurred. Therefore, SFF relies on a sharpness operator to find the depth where each point appears the more sharp, and reconstructs a depth image: In the absence of regularization (*blind* estimation), the depth of each pixel of the 2D scene maximizes the pixel's sharpness measure. Specifically, [Nayar and Nakagawa \[1994\]](#) approximates the sharpness curve (that represents the sharpness values versus the focus parameter values) with a Gaussian model, and interpolates (along the optical axis) the three focus measures centered on the maximum sharpness value to allow for a better depth estimation. To reduce the sensitivity to noise, some authors do the sharpness curve interpolation by using quadratic, cubic or polynomial interpolation [Moeller et al. \[2015\]](#) or Gaussian interpolation [Ribal et al. \[2018\]](#).

Nevertheless, *blind* depth estimation remains prone to noise and ambiguities since, in homogeneous or poorly textured areas, the measured sharpness will be quite low and unreliable. To overcome this limitation, some authors [Gaganov and Ignatenko \[2009\]](#); [Moeller et al. \[2015\]](#); [Ribal et al. \[2018\]](#) considered SFF in the variational framework. With regularization, the information extracted in the scene areas where SFF is reliable, such as in objects details, contours, is propagated from neighbors to ambiguous areas, such as homogeneous, overexposed or underexposed regions. As stated in Section C.1, to overcome the alteration of thin structures, there is however a need for anisotropic regularization, all the more critical that we work with superpixels.

C.2.2 From pixel level to superpixel one

Let us consider the 3D space defined by an orthonormal basis $(\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2)$ such that \mathbf{e}_0 and \mathbf{e}_1 are aligned with the image row and column dimensions and \mathbf{e}_2 will represent the focus dimension. The set of input image pixels, denoted \mathcal{P} , defines a cube in $(\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2)$ having dimensions $n_{\text{row}} \times n_{\text{col}} \times n_{\text{foc}}$, where n_{row} , n_{col} , and n_{foc} are positive integers. We also assume without loss of generality that these three dimensions are sampled with a unit step even if for focus it implies a simple transformation.

Then, we assume that superpixels are invariant

with focus dimension and we thus define them equivalently in 2D ($(\mathbf{e}_0, \mathbf{e}_1)$ plane) or 3D by replicating them along the focus dimension (\mathbf{e}_2). In the following, we denote by \mathcal{S} the set of superpixels defined in 2D and by \mathcal{S}^{13} the set of superpixels extended to 3D.

In this study, we aim at extending SFF variational formulation to superpixel level. However, the sharpness profiles and the superpixels themselves have to be preliminary computed at pixel level. Specifically, any sharpness value shall be computed at pixel level by nature. A sharpness operator is a function computed at every pixel $p \in \mathcal{P}$ and a sharpness profile is a vector gathering the sharpness values obtained by varying the focus dimension for a given pair of (row, column) coordinates in $(\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2)$. In the following, we denote by $f(p)$ the sharpness operator defined in pixel $p \in \mathcal{P}$ and by \cdot_{\downarrow} the projection on $(\mathbf{e}_0, \mathbf{e}_1)$ such that \mathcal{P}_{\downarrow} and p_{\downarrow} are the projections of \mathcal{P} and p respectively, and $\mathbf{f}(p_{\downarrow})$ denotes the sharpness profile at any pixel $p_{\downarrow} \in \mathcal{P}_{\downarrow}$. Then, the maximum of sharpness is estimated at any $p_{\downarrow} \in \mathcal{P}_{\downarrow}$ as $\max_{k \in [1, n_{\text{foc}}]} \mathbf{f}_k(p_{\downarrow})$ where \mathbf{f}_k denote the k^{th} component of sharpness profile \mathbf{f} . From maximum of sharpness, one can estimate the all-in-focus image defined on \mathcal{P}_{\downarrow} . This image allows us to compute the superpixels, \mathcal{S} , that have a good sensitivity to the sharp edges of the scene, since in 2D space it picks the pixel that is the “sharpest”, i.e. that has the highest contrast with its neighbors. The chance of constructing superpixels on blurred edges of the objects is minimized this way. From \mathcal{S} , the set of superpixels extended to 3D \mathcal{S}^{13} is then derived by duplicating n_{foc} times any superpixel $s \in \mathcal{S}$ along the axis \mathbf{e}_3 .

The sharpness values of \mathcal{S}^{13} elements can then be derived from the mean sharpness values of the 3D pixels $p \in \mathcal{P}$ that compose it, and, for each superpixel in \mathcal{S} , a *blind* depth estimation is derived from the maximum of sharpness varying focus (\mathcal{S}^{13} elements derived from a given superpixel $s \in \mathcal{S}$). In the following, the *blind* depth superpixel map is denoted $\hat{\mathbf{u}} = (\hat{u}_s)_{s \in \mathcal{S}}$ with $\hat{u}_s \in \mathbb{N}$ assuming (without loss of generality) that depth values are sampled as integer numbers. This depth map may be noisy and sensitive to the low sharpness profile of homogeneous regions of the scene, which we cope with our anisotropic neighborhood based regularization.

Finally, let us specify that, in our case, we consider the sharpness operator introduced in [Pertuz et al. \[2013\]](#), namely the Summed Modified LAPlacian (SMLAP): $f(p) = \text{SMLAP}(p), \forall p \in \mathcal{P}$.

C.2.3 Energetic formulation

Usual energetic formulations map a realisation of the random field we search, i.e. \mathbf{u} in SFF application, to a real number representing its inadequacy to correspond to the observations and prior knowledge. In our case, since the neighborhoods are also unknown, we made the energy depend also on a neighborhood field that maps a local anisotropic neighborhood to any field element (superpixels in our case). In the following, $\mathbf{u} \in \mathbb{N}^{\mathcal{S}}$ is the researched depth field, \mathbf{V} is the neighborhood field and \mathbb{V} is the set of possible neighborhood fields. Then, we aim at finding a minimizer of

$$F(\mathbf{u}, \mathbf{V}) = E_1(\mathbf{u}) + \alpha E_2(\mathbf{u}, \mathbf{V}), \quad (\text{C.1})$$

where $\alpha \in \mathbb{R}_{\geq 0}$ is a hyperparameter that need to be later tuned by the user. Specifically, the data fidelity term $E_1(\mathbf{u})$ is instantiated with a quadratic distance to the *blind* estimate $\hat{\mathbf{u}}_s$:

$$E_1(\mathbf{u}) = \sum_{s \in \mathcal{S}} W_s (u_s - \hat{u}_s)^2, \quad (\text{C.2})$$

where W_s depends on the dynamics of the sharpness profile normalized by its averaged value:

$$W_s \propto \left(\frac{\max_{k \in [1, n_{\text{foc}}]} (\mathbf{f}_k(s)) - \min_{k \in [1, n_{\text{foc}}]} (\mathbf{f}_k(s))}{\frac{1}{n_{\text{foc}}} \left(\sum_{k \in [1, n_{\text{foc}}]} \mathbf{f}_k(s) \right) - \min_{k \in [1, n_{\text{foc}}]} (\mathbf{f}_k(s)) + \epsilon} \right), \quad (\text{C.3})$$

with $\epsilon \in \mathbb{R}_{> 0}$ a small positive real number. With the weighting term W_s , the importance of the data fidelity term E_1 is decreased when the sharpness profile is homogeneous or when it presents a very low dynamic. Conversely, the areas with a sharpness profile with a precisely localized high response have high values of W_s reflecting the belief that they are trustful.

The regularization term $E_2(\mathbf{u}, \mathbf{V})$ is derived from the TV operator. For any $\mathbf{u} \in \mathbb{N}^{\mathcal{S}}$ and any $\mathbf{V} \in \mathbb{V}$, it is defined as

$$E_2(\mathbf{u}, \mathbf{V}) = \sum_{s \in \mathcal{S}} \sum_{t \in \mathbb{V}(s)} W_{st} |u_s - u_t|, \quad (\text{C.4})$$

where W_{st} is a weighting function depending on the neighborhood field \mathbf{V} :

$$W_{st} = \frac{1}{2} \left(\frac{1}{\sharp \mathbb{V}(s)} + \frac{1}{\sharp \mathbb{V}(t)} \right), \quad (\text{C.5})$$

where $\sharp \mathbb{V}(s)$ denotes the cardinality of the neighborhood at the superpixel s . The weighting term W_{st} aims at normalizing the regularization terms E_2 with respect to the size of the considered neighborhoods since this latter is no longer constant (as it was with usual 4 or 8-connectivity for instance at pixel level).

C.2.4 Optimization

Graph cuts optimization refers to the computation of minimum cuts/maximum-flows in a graph of appropriate topology for minimizing functionals arising in computer vision, e.g. composed of unary and pairwise terms. Compared to other combinatorial algorithms, graph cuts are very competitive both in terms of accuracy (global minimum is very well approached if not reached as for many binary problems) and running time (by avoiding stochastic iterative convergence) for a wide range of computer vision tasks [Szeliski et al. \[2008\]](#). Practically, graph cuts depict linear complexity in the number of sites of \mathcal{S} [Boykov and Kolmogorov \[2004\]](#). Moreover, compared to continuous minimization algorithms, they are able to deal with regular or irregular lattices without any difficulties.

In the binary case, the key idea of graph cuts is to construct a two-terminals graph, where nodes are sites of \mathcal{S} and edges encode relationships between nodes, in a such a way that any cut separating these terminals is equal to the value of the functional on the underlying binary labeling. In particular, when all pairwise terms are submodular, polynomial-time maximum-flow algorithms allow for efficiently finding the minimum-cut in a graph and thus a global minimizer of the functional for binary problems [Boykov and Kolmogorov \[2004\]](#).

In the multi-labels case (such as in our case), efficient algorithmic schemes exist for finding minimizers of functionals. As explained in Section C.3, we intend as a first attempt in this paper to minimize F (see Equation (C.1)) with V fixed. It is not difficult to see that the functional $u \mapsto F(u, V)$ is convex based on Equation (C.2), (C.3), (C.4) and (C.5). In such a situation, a global minimizer of this functional can be efficiently obtained by decomposing the problem into a set of subproblems only involving binary variables (as in the case of isotropic neighborhoods in [Ribal et al. \[2018\]](#)), where each one of them is solved standard graph cuts in the aforementioned binary case.

C.3 Anisotropic neighborhood construction

The construction of anisotropic neighborhoods can be decomposed into (i) the estimation of the presence of thin structures (in our case performed by a vesselness operator) discussed in Sections C.3.1 and C.3.2, and (ii) the actual computation of the neighbors of each superpixel discussed in Section C.3.3. Let us

recall that the neighborhoods are constructed on an irregular lattice of superpixels, and that a neighborhood relationship can be formalized on a graph representing the superpixels by vertices, by the edges interconnecting some vertices. The neighborhood is thus an application that maps the set of superpixels \mathcal{S} to its powerset $2^{\mathcal{S}}$ without any specific constraint (e.g., bound on spatial distance) at this stage. We want to outline that it may thus be different to the notion of *adjacency* that refers to the existence of a common border between the superpixels and that allows for the definition of connected components.

Then, to estimate anisotropic neighborhoods, we will rely on a guidance map, denoted \mathbf{g} , that encodes the information of anisotropy and orientation for every superpixel $s \in \mathcal{S}$. Such a map must encourage the alignment of neighborhoods with the thin structures of the image. In the absence of knowledge of the scene objects, the estimation of \mathbf{g} is not trivial at all. Indeed, considering simultaneous estimation of \mathbf{g} and the segmentation or reconstruction, the resolution appears very complex if not intractable, and considering alternate estimation would require an iterative scheme ensuring the convergence in a controlled number of iterations. Therefore in this study, we rather focus on a single estimation of \mathbf{g} as a first attempt, with obvious methodological and computational benefit, at the expense of defining an estimation sufficiently robust to the input data imperfections to yield some trustworthy guidance map. Simultaneous estimation of \mathbf{u} and V is left for future work. More specifically, if the estimation of \mathbf{g} bases on local estimates, its construction must be robust to noise in these latter. We investigate two options, the *Tensor Voting* (TVo) as presented by [Medioni et al. \[2000\]](#) and the *Ranking the Orientation Responses of Path Operators* (RORPO) vesselness operator as introduced by [Merveille et al. \[2018\]](#). In what follows, \mathbf{g} is a field of \mathbb{R}^2 vectors encoding both the direction and the saliency.

C.3.1 Tensor Voting-based guidance map

Tensor Voting basics

Tensor Voting (TVo) has been selected for its robustness to noise and efficiency for connecting thin structures like edges [Medioni et al. \[2000\]](#). TVo relies on the Gestalt principles of perceptual organization (such as proximity, continuity and similarity) for designing the voting operation. Its formulation involves one scale parameter, $\sigma_T \in \mathbb{R}_{>0}$, setting the spatial range in which most of the energy of the TVo will be distributed. The basic idea is that casting a vote to other

site locations allows the information of each tensor to be propagated, and then thanks to the voting step, the tensors are smoothed and their orientations refined. Voting operation is performed through voting kernels that have continuous and smoothly varying orientations of eigenvectors and decreasing eigenvalues, except at the origin of the kernel. For implementation purpose, the voting kernels are often discretized and stored into a precomputed field of tensors, which evaluates the values of the tensors cast from the voter on each point of a regular lattice. Appendix B.2.1 specifies and gathers all the main equations useful for 3D TVo, that is much more complex than 2D one used in Zou et al. [2012] for instance.

In Medioni et al. [2000], TVo involves the five following main steps. First step is the initial vote that requires the definition of the initial set of voters also called *tokens* and the set of cast locations (for vote). In the absence of orientation information, the set of tokens is usually converted into a sparse set of ball tensors that vote in every image site. Second step is a refinement step. Based on the previous sparse vote, the initial set of ball tensors can be refined into a set of stick tensors. For this, each tensor is projected on the stick tensor axis in the basis used for tensor decomposition. Third step is a dense voting in order to propagate the stick information at every point. It yields the dense tensor map. Then, fourth step projects the tensors on the three axes of the decomposition basis so that three saliency maps can be derived, encoding for surface, curve and junction saliency. From these maps, the final step of the algorithm derives the probabilities of presence of surfaces, curves and points.

Computation of guidance map

We adapt TVo to our SFF problem as follows. The *tokens* are the local maxima of sharpness profiles in every superpixel (in $(\mathbf{e}_0, \mathbf{e}_1)$ plane). To avoid redundancy between close maxima (inducing artificial reinforcement of these latter) of a same profile, a non-maximum suppression step is performed on sharpness profiles: Specifically, we only keep one maximum per continuous interval of focus values associated to sharpness values greater than 80% of the maximum sharpness. This way, we ensure that the *tokens* are all separated by a local minimum having value below 80% of the global maximum. This initialization provides a tensor map that is sparse in 3D, but dense in 2D.

Then, since the number of pairs in $(\mathcal{S}^{13} \times \mathcal{S}^{13})$ is very large, the vote for the orientation estimation is also restricted to the set of *tokens*. This allows for

reducing the computational burden by removing the dense voting step, at the risk of a loss of accuracy when the initial depth estimations (and thus *tokens*) are erroneous.

Although TVo allows us to handle tensors defined in \mathbb{R}^3 for the vote, at the end (for decision after voting) we have to decide a single tensor for any 2D superpixel $s \in \mathcal{S}$. In our experiments, we found that the most convincing results are obtained when only considering the cumulated tensors (after voting) at the *blind* estimated depth \hat{u}_s . Indeed, while this leads to irregularities when $\hat{\mathbf{u}}$ is noisy, this also allows for gaps in the orientations estimated on the edges of the structures of the images, which could be beneficial. Then, for extracting the guidance map \mathbf{g} , for each superpixel $s \in \mathcal{S}$, we project the selected tensor (in \hat{u}_s) into image plane and derive the major eigenvector $\hat{\mathbf{e}}_{0s}$ in s and the two eigenvectors $(\lambda_{0s}, \lambda_{1s}) \in \mathbb{R}_{\geq 0}^2$ so that the saliency and orientation of the guidance map in s is computed as follows:

$$\mathbf{g}_s = (\lambda_{0s} - \lambda_{1s})\hat{\mathbf{e}}_{0s}, \quad \forall s \in \mathcal{S}.$$

C.3.2 RORPO-based guidance map

As an alternative to TVo, we consider RORPO, a non linear operator based on mathematical morphology and used for thin structure detection (see Merveille et al. [2018] for more details).

RORPO basics

The idea of RORPO is to use a set of oriented filters with different orientations to analyze the image in terms of the response of multiple morphological operations. Indeed, for a thin structure, at least one dimension is substantially smaller than the other ones by definition. Thus, determining the image areas where only a small number of high responses are measured among the oriented filters discriminates the thin structures. These oriented filters, called path openings, are parameterized by structuring functions defining the set of connection relationships \mathcal{R} between sites (pixels, or superpixels in our case). Since the RORPO is based on these path openings, let us briefly recall how these latter work, firstly on a binary image and secondly on a gray level image.

Denoting by \mathcal{R} the set of connection relationships, for each connection relationship $\rightsquigarrow_{\theta} \in \mathcal{R}$, we remind how a path opening is defined for binary images in Merveille et al. [2018]: Given $\rightsquigarrow_{\theta}$ and a length $L \in \mathbb{R}_{>0}$, the path opening $\mathcal{O}_{\rightsquigarrow_{\theta}, L}(X)$ is the union of all paths connected by $\rightsquigarrow_{\theta}$ and of length L in the set of

true pixels (1-valued) in the considered binary image X . Each path opening filters out the structures that are not aligned with the considered orientation. Thus, a thin structure will be deleted by at least one oriented filter, conversely to isotropic structures that will have an homogeneous answer to the set of path openings.

Then, to extend binary path openings to gray level images, one considers level sets, i.e. sets of sites having a value greater than the considered gray level: Given \rightsquigarrow_θ and a length $L \in \mathbb{R}_{>0}$, the gray level path opening of an image Y is defined as

$$\mathbb{O}_{\rightsquigarrow_\theta, L}(Y, s) = \max \{ \tau \in \mathbb{R}_{>0} \mid s \in \mathbb{O}_{\rightsquigarrow_\theta, L}(Y_{\geq \tau}) \},$$

where $Y_{\geq \tau}$ is the level set of Y at level τ .

In [Merveille et al. \[2018\]](#), RORPO implementation involves the following five main steps. The first step is the dilation of the gray level input image with respect to spatial adjacency. The second step deals with direction sampling. It boils down defining a finite set of connection relationships, denoted by \rightsquigarrow_θ , such that two sites s and t are connected if and only if (i) they are adjacent and (ii) \overrightarrow{st} vector's direction and the sampled direction θ are considered equal been given the imprecision angle ϕ_T threshold. The third step is the computation of the path opening results for the sampled directions and the fourth step ranks their responses as follows: For each site, the responses to the $\#\mathcal{R}$ path openings are ranked in decreasing order of magnitude, i.e. denoting RF_1 the maximum value and $RF_{\#\mathcal{R}}$ the minimum (last in the ordering) value. This ranking of the orientation responses of the path openings gave its name to the algorithm RORPO. Then, for each site, the RORPO value is the difference between maximum path opening value (RF_1) and the i largest response, (RF_i). In our case, we set $i = 4$. Finally, fifth step derives, for each site, an orientation by averaging the orientations of the three largest responses.

This formulation yields higher responses for thin structures that have a small number of high responses in path openings. Therefore, the value returned by the RORPO allows us to discriminate the saliency of thin structures. Let us now present the estimation of orientations used to derive the guidance map \mathbf{g} .

Computation of guidance map

Like TVo, our implementation of RORPO works with the data volume corresponding to the sharpness profiles in every superpixel. The choice of these input data is fully relevant for the RORPO that will detect structures presenting high gray level values and indeed we want to detect highest sharpness values.

For numerical convenience, path openings are only performed with 2D slices, i.e. at given focus value, which boils down researching structures in image plane. This is simply performed by restricting the connection relationships \rightsquigarrow_θ to be within image plane. Then, we consider six directions \mathbf{v}_θ in the image plane, characterized by their positive angle θ with the \mathbf{e}_0 axis: $\mathbf{v}_\theta = \cos(\theta)\mathbf{e}_0 + \sin(\theta)\mathbf{e}_1$ (see Figure C.1).

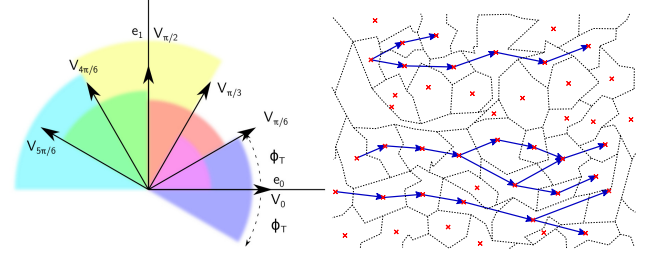


Figure C.1 – Illustration of the 6 directions of \mathcal{R} (left) and an example of path obtained with one structuring function \rightsquigarrow_θ (right). The connectedness \rightsquigarrow_θ is characterized by the vector \mathbf{v}_θ and the angular width ϕ_T . For this illustration, we have represented directed edges for positive displacements, but the paths are computed in both directions.

Note that, although usually the length L is a positive integer expressed in pixel unit, extending the case of pixel lattice to superpixel one, we instead consider that the length of the path is a real $L \in \mathbb{R}_{>0}$, computed as the sum of the distances between the superpixels' barycenters in the path.

Each of the connection relationships yields a path opening result. From this set of path openings, we firstly compute the RORPO index that is further interpreted as a saliency index and secondly the structure orientation. For the latter, we use a specific average operation such that orthogonal vectors cancel and vectors of opposite directions would not. The trick consists in considering polar coordinates and doubling the argument value of the vectors before averaging them, and dividing the argument of the averaged result by two. Mathematically, with complex notations, and omitting the normalization coefficient useless here, it is as follows:

$$\mathbf{v}_{RO}(s) \propto \left(\epsilon + \sum_{\rightsquigarrow_\theta \in \mathcal{R}'(s)} \mathbb{O}_{\rightsquigarrow_\theta, L}(Y, s) \exp(2i\theta) \right)^{\frac{1}{2}},$$

where $\epsilon > 0$ is a very small real number used for numerical stability of the expression, and $\mathcal{R}'(s) \subset \mathcal{R}$ is the set of orientations of the three first answers in the rank filter at site s .

With this construction, we obtain a 3D volume of vectors that has the same dimensions as the grayscale input data and from which the 2D guidance map \mathbf{g} is

finally computed. For this, we average the \mathbf{v}_{RO} directions varying the depth:

$$\mathbf{g}_s = \left(\sum_{t \in \mathcal{S}^{13}, t_l = s} |\mathbf{g}_t| \exp(2i \arg(\mathbf{g}_t)) \right)^{\frac{1}{2}}, \quad \forall s \in \mathcal{S},$$

where t_l is the result of the projection of 3D site t on the 2D image plane and $\arg(c)$ denotes the argument of any complex number $c \in \mathbb{C}$. In previous equation, the angles are simply weighted by the norm \mathbf{g}_t , but more sophisticated weighting may also consider the distance with respect to the *blind* depth, e.g. using a weighting coefficient equal to $|\mathbf{g}_t| \exp(-\frac{2(t-\hat{u}_s)^2}{\Delta_h})$.

Compared to TVo, RORPO allows for a faster computation of the guidance map and is consistent with the notion of path-based neighborhood introduced in Section C.3.3 that specify the construction of the neighborhoods from \mathbf{g} .

C.3.3 Path-based neighborhoods

This section depicts our contribution concerning the construction of anisotropic neighborhoods, i.e. the neighborhood field $V \in \mathbb{V}$ (see Section C.2.3).

We propose path-based neighborhoods to fit into thin structures of the image, possibly one superpixel width. Being based on the adjacency graph \mathcal{A} , the neighborhood construction ensures that the neighbors of a superpixel defines a single connected component. We recall that two superpixels are adjacent when they share a common border at pixel level, thus the adjacency is a symmetric relationship: $s \in \mathcal{A}(t) \iff t \in \mathcal{A}(s), \forall s, t \in \mathcal{S}$. Then, a path of length $n \in \mathbb{N}$ is an ordered list (s_0, \dots, s_n) of consecutive adjacent superpixels.

More formally, let us denote by $\Pi_K(s, t)$ the set of paths joining any pair of superpixels $(s, t) \in \mathcal{S}^2$, without any loop, and having length K : $\Pi_K(s, t) = (s_0, \dots, s_K) \subset \mathcal{S}^{K+1}$, such that $\forall k \in \llbracket 0, K \rrbracket, s_{k+1} \in \mathcal{A}(s_k), s_0 = s, s_K = t$, and $\forall j, k \in \llbracket 0, K \rrbracket, s_j \neq s_k$. Similarly, we also define $\Pi(s, t)$, the set of paths joining superpixels s, t with any length, by extension:

$$\Pi(s, t) = \bigcup_{K \in \mathbb{N}} \Pi_K(s, t).$$

The proposed path-based neighborhoods relies on previously estimated guidance map \mathbf{g} that contains the information about the orientation and saliency of the structures of the scene, useful to define neighborhoods in every superpixel. In particular, when the norm $\|\mathbf{g}_s\|$ at a given superpixel s is below a fixed threshold, the neighborhood in s is $V(s) =$

$\mathcal{A}(s)$, i.e. an isotropic neighborhood that ensures adjacency. Otherwise, the set of neighbors is given by the union of the elements of two paths that expand from s to the two opposite directions corresponding to the orientation of \mathbf{g}_s . In the next subsections, we present the two options investigated for constructing the path-based neighborhoods.

Target-based neighborhood

Target-Based Neighborhood (TBN) is derived from paths that join, starting from a source superpixel $s \in \mathcal{S}$, two "targets" corresponding to distant superpixels $(t_0^*, t_1^*) \in (\mathcal{S} \setminus \{s\})^2$. Specifically, for $j \in \{0, 1\}$, the targets are selected with

$$t_j^* \in \underset{t \in \mathcal{S} \text{ s.t. } (-1)^j \langle \mathbf{g}_s, \vec{s}t \rangle > 0}{\operatorname{argmin}} \quad \|I(s) - I(t)\|_2^2 - \eta \|\vec{s}t\|_2 \times \left| \langle \mathbf{g}_s, \vec{s}t \rangle \right|, \quad (\text{C.6})$$

where $\langle \cdot, \cdot \rangle$ denotes the dot product between two vectors so that the constraint $(-1)^j \langle \mathbf{g}_s, \vec{s}t \rangle > 0$ refers to an half-space domain, \leq, \cdot, \geq stands for the cosine similarity (also called normalized dot product) between two vectors, $\|\cdot\|_2$ is the Euclidean norm of a vector, $|\cdot|$ is the absolute value of a real number, and $\eta \in \mathbb{R}_{>0}$ an hyperparameter to set.

In Equation (C.6), the first term favors the superpixels s and t to share similar image intensities while the second one favors far targets being aligned with \mathbf{g}_s . Note that, for selecting close neighbors, the range of search is restricted to an ellipse centered at s with major axis aligned with \mathbf{g}_s and that solutions are derived by Dynamic Programming (DP).

Then, the paths are selected among the two sets $\Pi(s, t_0^*)$ and $\Pi(s, t_1^*)$ joining s to t_0^* and t_1^* , respectively. For doing so, we formulate a cost function that the optimal path (denoted by p_j^* , $j \in \{0, 1\}$, has to minimize:

$$p_j^* \in \underset{p \in \Pi(s, t_j^*)}{\operatorname{argmin}} \quad \sum_{k=0}^{|p|-1} \|I(p(k)) - I(p(k+1))\|_2^2, \quad (\text{C.7})$$

where $|p|$ stands for the length of the path p , and $p(k)$ denotes the k^{th} element of it. The term to minimize in Equation (C.7) is large when the gray levels of successive superpixels along a path are dissimilar and small otherwise. We also use DP to derive a minimizer of previous equation, and the neighborhood $V(s)$ is finally constructed as the set of the superpixels in p_0^* or in p_1^* , but s : $V(s) = (p_0^* \cup p_1^*) \setminus \{s\}$. The adjacency along these paths being ensured by construction, the derived neighborhood forms a single connected component.

We give an illustration of this kind of neighborhood in Figure C.2a. While this neighborhood ensures that the set of neighbors of a superpixel s forms a single connected component and favors paths oriented in the estimated direction of thin structures, there is no guarantee concerning the cardinality of these paths. Depending on the image content that influences the location of the target sites, one site may have a very small amount of neighbors in a very contrasted location, or conversely could possibly have a large number of neighbors if there exists an arbitrary long path with constant radiometry. In our experiments, we set $\eta = 30$.

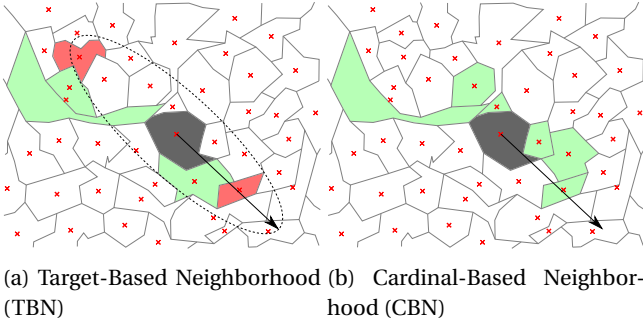


Figure C.2 – Toy example of path-based neighborhoods. The arrow indicates the orientation of \mathbf{g}_s and superpixels of neighborhood are shown in color. For TBN, two “targets” (in red) are selected in an ellipse centered on the source superpixel s (in grey). For CBN, two paths with $K = 3$ elements are built to be aligned with \mathbf{g}_s , taking into account radiometric similarity with the site s .

Cardinal-based neighborhood

Cardinal-Based Neighborhood (CBN) is also a path-based neighborhood. However, instead of constraining the path extremities like TBN (see Section C.3.3), it constraints path cardinality (and thus neighborhood cardinality): $\forall s \in \mathcal{S}$, $V(s)$ is the union (excepting element s) of two length-fixed paths $p_0^*, p_1^* \in \Pi_K(s, \cdot)$, with $\Pi_K(s, \cdot)$ denoting the set of paths of length $K \in \mathbb{N}_{>1}$ starting from s . Additionally, these paths are encouraged to expand in opposite directions.

As previously, we define a cost function presenting a tradeoff between fidelity to the thin structure orientation and fidelity to the gray level of originating superpixel s . For any $j \in \{0, 1\}$,

$$p_j^* \in \operatorname{argmin}_{p \in \Pi_K(s, \cdot)} \sum_{t \in p} \|I(s) - I(t)\|_2^2 + \eta' \psi_j(\vec{st}, \mathbf{g}_s),$$

where $\eta' \in \mathbb{R}_{>0}$ is an hyperparameter to set, and

$$\psi_j(\vec{u}, \vec{v}) = \begin{cases} \arccos(|\langle \vec{u}, \vec{v} \rangle|) & \text{if } (-1)^j \langle \vec{u}, \vec{v} \rangle > 0, \\ +\infty & \text{otherwise,} \end{cases}$$

measures the angle between the vectors \vec{u} and \vec{v} and discriminates whether the dot product is positive or not.

Note that, in CBN, the cost function compares gray level and positions of each site of the path versus the site s instead of computing these differences on the adjacent sites on the path (as with TBN), to allow local deviations while ensuring global neighborhood orientation and gray level value. Figure C.2b shows an example of neighborhood constructed with such an approach. In our experiments, we set $\eta' = 100$ and $K = 3$.

C.4 Experiments and results

C.4.1 Data

We test our approach on some scenes extracted from the public¹ Middlebury College dataset [Scharstein and Pal \[2007\]](#). For each scene, a ground truth depth map and an all-in-focus RGB image are provided. Both images have 360×360 pixels. These images enable us to simulate the desired set of blurred images thanks to the defocusing algorithm [Pertuz et al. \[2013\]](#), that is currently available as a Matlab source on MathWorks file exchange. In our simulations, the multiple images correspond to different focal object plane depths along the axis \mathbf{e}_2 (whereas $(\mathbf{e}_0, \mathbf{e}_1)$ is a basis for image plane). For simplicity and readability, we simulate images at focus values regularly sampled and set this step to be the unit. The maximum depth, denoted by $\Delta_h \in \mathbb{N}$, is therefore equal to n_{foc} that we set equal to 50. However, images taken at irregular steps could be considered as well without loss of generality.

Then, the set of defocused images is assumed to be the only input data available, and we reconstruct depth values based on the following steps. Firstly, we compute the sharpness profiles in each pixel independently and from maximum of these profiles, we derive the *blind* estimate of all-in-focus image. Secondly, we compute the superpixels from this *blind* all-in-focus image. The number of superpixel algorithms proposed in the literature is rather important, including different kinds of superpixels that embed different properties, such as the adherence to the boundaries of the objects, the compactness or convexity of the resulting superpixels, their regularity, or the smoothness of their boundaries. We refer the reader to [Stutz](#)

¹<https://vision.middlebury.edu/stereo/data/>

et al. [2018] to have an overview of the variety of superpixel algorithms. In practice, after a few comparisons, we focus on the superpixels provided by an algorithm called ETPS Yao et al. [2015], since it is energy based (as the general framework adopted for our work) and offers relatively smooth and regular superpixels. Thirdly, as described in Section C.2.2, the sharpness profile in each superpixel as well as the *blind* superpixel depth map $\hat{\mathbf{u}}$ are derived. Fourthly, we compute the guidance map \mathbf{g} and construct the neighborhood field $\{V(s), \forall s \in \mathcal{S}\}$, based on the chosen method as described in Sections C.3.1, C.3.2, C.3.3 and C.3.3. Fifthly, V and $\hat{\mathbf{u}}$ allow us to instantiate our anisotropic regularization and to derive the regularized depth map results presented in the following next sections.

C.4.2 Evaluation criteria

The Ground Truth (GT) provided in Middlebury College dataset Scharstein and Pal [2007] is at pixel level. To perform evaluation, we duplicate the depth estimated for a given superpixel to each of its pixels. In the following, for the sake of clarity, we prefer not to change the variable name so that we also denote by \mathbf{u} the estimated depth map at pixel level (the element lattice \mathcal{P} or \mathcal{S} removing ambiguity if any) and by $\tilde{\mathbf{u}}$ the GT.

Evaluation metrics We focus on three complementary global metrics, namely RMSE (Root Mean Square Error) that has good additive properties, PSNR (Peak Signal to Noise ratio) derived from RMSE and SSIM (Structural Similarity Index Measure Wang and Sheikh [2004]) that bases on perception-model to measure the similarity between two images. The mathematical expressions of these metrics are as follows:

$$\text{RMSE}(\mathbf{u}, \tilde{\mathbf{u}}) = \sqrt{\frac{1}{\#\mathcal{P}} \sum_{p \in \mathcal{P}} (u_p - \tilde{u}_p)^2},$$

$$\text{PSNR}(\mathbf{u}, \tilde{\mathbf{u}}) = 20 \log_{10} \left(\frac{\Delta_h}{\text{RMSE}(\mathbf{u}, \tilde{\mathbf{u}})} \right),$$

$$\text{SSIM}_{\Omega}(\mathbf{u}, \tilde{\mathbf{u}}) = \frac{1}{\#\mathcal{P}} \sum_{p \in \mathcal{P}} \frac{(2\mu_{u,p}\mu_{\tilde{u},p} + C_1)(2\sigma_{u,\tilde{u},p} + C_2)}{(\mu_{u,p}^2 + \mu_{\tilde{u},p}^2 + C_1)(\sigma_{u,p}^2 + \sigma_{\tilde{u},p}^2 + C_2)},$$

where $\#\mathcal{P}$ stands for the cardinality of \mathcal{P} , Ω is a window centered at any pixel p and of size 7×7 in our case, $\mu_{u,p}$, $\mu_{\tilde{u},p}$ are the means over Ω centered at p of \mathbf{u} and $\tilde{\mathbf{u}}$ values respectively, $\sigma_{u,p}^2$, $\sigma_{\tilde{u},p}^2$, and $\sigma_{u,\tilde{u},p}$ are the variances and covariance, respectively. Finally, the constants C_1 and C_2 are computed from Δ_h as

$C_1 = (0.01\Delta_h)^2$ and $C_2 = (0.03\Delta_h)^2$ for numerical stability. This is the version of SSIM specified in Wang and Sheikh [2004] with (according to their notations) $\alpha = \beta = \gamma = 1$. By computing the variances, covariance and mean values on a set of windows covering the whole image, SSIM incorporates comparison measurements of luminance, contrast and structure of images that allows to take into account important perceptual phenomena in its evaluation.

For result comparison, we remind that the lower the RMSE values are (in $[0, \Delta_h]$), the better the results are while for PSNR and SSIM criteria, higher values (in $\mathbb{R}_{\geq 0}$ and $[0, 1]$ respectively) reflect better performance.

Evaluation maps Three complementary kinds of maps allow us to visualize the difficult areas. Firstly, depth error map, called E , will stress the image areas with poorest reconstruction. Secondly, neighborhood orientation map will represent saliency and direction information extracted from the guidance map, that allows us to evaluate qualitatively this latter. Thirdly, depth dynamic within neighborhoods, called Q_V , provides a measure of the neighborhood consistency in terms of depth. Pixel values of E and Q_V maps are computed as follows:

$$E(p) = |u_p - \tilde{u}_p|, \quad \forall p \in \mathcal{P},$$

$$Q_V(p) = \max_{q \in V(p)} |\tilde{u}_q - \tilde{u}_p|, \quad \forall p \in \mathcal{P},$$

where $V(p)$ at pixel level is simply the set of pixels that belong to any superpixel neighbors of the superpixel including p .

Concerning the interpretation of these maps, the lower the E values (in $[0, \Delta_h]$), the better the depth estimation at considered pixel. The orientation map is expected to be relatively smooth while following the sharp edges of the objects and aligning with the thin structures. Finally, in Q_V , low values (in $[0, \Delta_h]$) reflect a consistent neighborhood (without implying uniqueness of the solution). Note that a major benefit of Q_V criterion is that it does not require any neighborhood ground truth (which we obviously do not have).

C.4.3 Alternative approaches considered for comparison

To evaluate the benefits of our approach compared against isotropic neighborhoods or simplest anisotropic ones, we focus on the following alternative approaches.

Stawiaski’s isotropic neighborhood In [Stawiaski and Decencière \[2011\]](#), an isotropic neighborhood is computed such that the superpixels that share a common border are neighbors and their interactions are weighted by the length of this common border. This neighborhood corresponds to the adjacency relationship, with a weighting function. This formulation ensures that the set constituted by a superpixel and its neighbors is a single connected component, but it does not ensure that the barycenters of neighboring superpixels are close from each other: Very large superpixels may therefore be included in the neighborhood of a given superpixel s while most of the pixels that constitute them are actually far from s . Fortunately, such configurations are rare for regular superpixels.

Shape-based neighborhood inspired from [Giraud et al. \[2017a\]](#) Shape-based neighborhood is an intuitive method for building anisotropic neighborhoods. The “shape” refers to the approximation of the neighborhood as a parametric shape, namely ellipse in our case. The neighbors of a superpixel s are then the superpixels whose barycenter is included in the “shape” centered in s . We considered in our case parametric ellipses whose major semi axis directions are given by the guidance map in s . Practically, when saliency in superpixel s is very low, i.e. $\|\mathbf{g}_s\|$ is lower a given threshold (0.05), the direction is not reliable so that we rather define neighborhood as a disc, which boils down to the isotropic superpatch neighborhood of [Giraud et al. \[2017a\]](#). Note that we do not exploit further saliency information that appears noisier than direction, and set the eccentricity as a constant parameter of the model.

Finally, let us underline that since such neighborhoods are computed from barycenters positions, they do not enforce adjacency of the neighbors. In particular, when the superpixels are highly irregular, concave, or with low compactness, they may yield neighborhoods with disconnected components. However, with compact, regular and convex superpixels, shape-based neighborhood provides an efficient and intuitive method for building anisotropic neighborhoods.

Perfect neighborhood For having an estimation of the possibly best performances brought by an anisotropic approach, we propose a “so-called” *perfect* anisotropic neighborhood. The latter is computed with respect to GT depth map $\bar{\mathbf{u}}$ as follows. *Perfect* neighborhood is implemented as a shape-based neighborhood with a disc of given radius centered in

$s \in \mathcal{S}$, where we remove the neighbors presenting a depth difference between the depths of GT and s higher than a fixed threshold $D_V = \frac{\Delta_h}{10} + 1$. Additionally, elements that do not belong to the s connected component are removed from the neighbors. Thus, *perfect* neighborhood refers to a neighborhood having good properties in terms of homogeneity, connectivity and shape, even if it is not unique.

C.4.4 Results

Global performance analysis

Let us first consider global performance obtained considering the whole Middlebury college dataset. Figure C.3 shows the results achieved using 5000 superpixels (ETPS [Yao et al. \[2015\]](#)), in terms of RMSE (allowing summing individual image performance), varying the regularization parameter α . We notice that the *perfect* neighborhood and the CBN, either from RORPO or TVo, yield the lowest RMSE values meaning they outperform all the other approaches for a wide range of regularization coefficients. Since *perfect* neighborhood was designed to evaluate the performance gain specifically related to anisotropic neighborhood (leaving apart the question of its estimation) with respect to isotropic one (represented by Stawiaski’s approach), the results clearly underline the benefit of anisotropic neighborhood for regularization. A satisfactory result is that CBN provides almost as good results as *perfect* neighborhood (which we remind is unrealistic since it requires GT), stressing the performance of neighborhood estimation itself. Comparing with “ellip” that refers to the “Shape-based neighborhood inspired from [Giraud et al. \[2017a\]](#)”, we notice that these latest results are much worse, underlining the importance of a fine (not too simplistic) estimation. About TBN estimation, we notice it only leads to interesting results for low regularization ($\alpha < 1$). Finally, we also note that RORPO or TVo use for \mathbf{g} estimation does not really impact the results, but a very slight advantage for RORPO. In conclusion, according to Figure C.3, best performance is achieved by RORPO CBN, with a very noticeable robustness of the results with respect to regularization coefficient, $\alpha \in [2, 16]$. This robustness of CBN to the regularization parameter that is also confirmed by visual inspection of error maps, is one of the strengths of this approach against its alternatives.

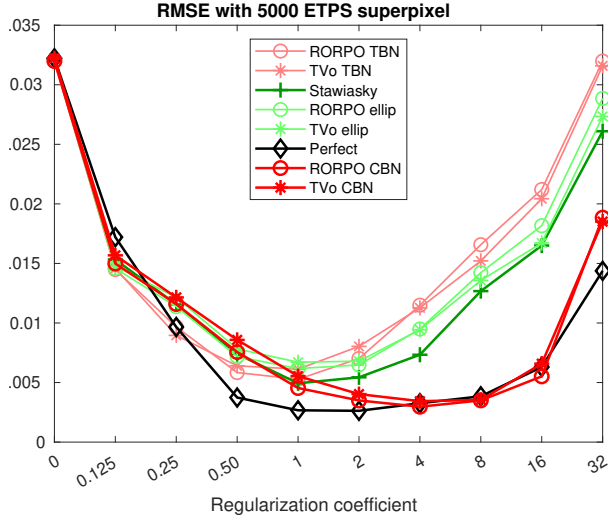


Figure C.3 – Comparison of neighborhood anisotropy benefit measured through RMSE on the whole dataset. The results are achieved using 5000 ETPS superpixels Yao et al. [2015] and different neighborhood estimations.

We now check the result dependency to superpixel segmentation. However, to investigate how results are dependent on ETPS superpixels, we consider, as an alternative to ETPS superpixels Yao et al. [2015], the WaterPixels (WP) proposed in Machairas et al. [2015]. Figure C.4 shows curves analogous to those in Figure C.3 considering either 5000 (like with ETPS superpixels) or 2000 WP, respectively. With respect to Figure C.3, we notice the curves and conclusions are remarkably similar, but a slight loss of performance when the number of superpixels is lower (it can be seen looking at the lowest value achieved considering *perfect* neighborhood) and a more distinct advantage for RORPO with respect to TVo (when looking at the CBN curves).

To further investigate the performance variability with respect to scene and/or superpixels, Figure C.5 and Figure C.6 respectively show the PSNR and the SSIM obtained on each scene, for the best result obtained with a varying α . First of all, the remarks concerning the robustness to the two kinds of considered superpixels (ETPS and WP) or their number (5000 and 2000) still hold: Difficult scenes are the same and CBN achieves very interesting performance in most cases. Indeed, on some scenes such as *Aloe1*, *Books1*, *Wood11*, all approaches yield equivalent results, whereas in other scenes such as *Lampshade*, *Plastic1* or *Reindeer*, achieved results appear more sensible to neighborhood estimation. We also note that the two criteria PSNR and SSIM are complementary since differences of performance can be visible in only one of them, such as with scene *Midd11* or *Moebius1*.

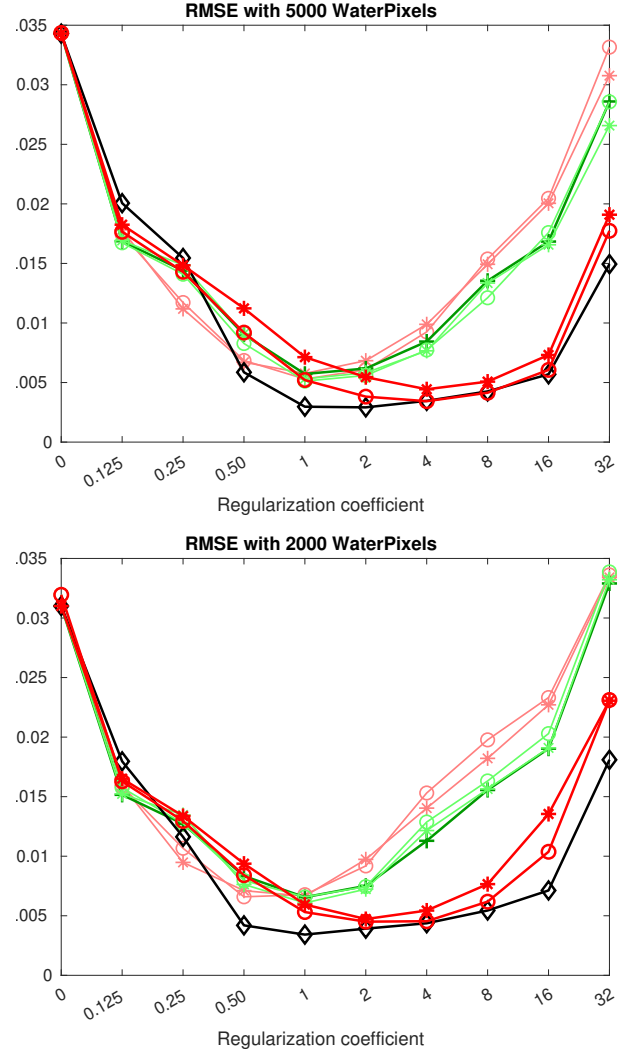


Figure C.4 – Comparison of neighborhood anisotropy benefit measured through RMSE on the whole dataset. The results are achieved using either 5000 WP (left) or 2000 WP (right) using different neighborhood estimations. The legend is the same as in Figure C.3.

However, let us underline that in most scenes, the top trio is RORPO-CBN, TVo-CBN and quite obviously *perfect* neighborhood. These approaches outperform both isotropic neighborhood (represented by Stawiaski's approach) and naive anisotropic one (ellipse-based). Nevertheless we also confirm the fact that an isotropic neighborhood assumption is preferable to too naive anisotropic neighborhood estimation. In conclusion, despite the scene disparity inducing variable performance, the main conclusions concerning the benefit of anisotropic neighborhood fine estimation can also be drawn at scene level.

Detailed analysis of two cases

For further analysis, we present the corresponding error maps and neighborhood quality maps, focus-

ing on some cases where the performance highly depends on the type of neighborhood, such as with the *Lampshade* scene and the *Reindeer* one.

Let us first consider the neighborhood estimation quality. As specified in Section C.4.2, the values of the depth dynamic within a neighborhood are in $[0, \Delta_h]$, with low values reflecting a consistent estimation of neighborhood. From Figure C.7, we clearly see that most of the heterogeneous neighborhoods are located at the borders of thin structures such as the lampshade rod or the reindeer antlers.

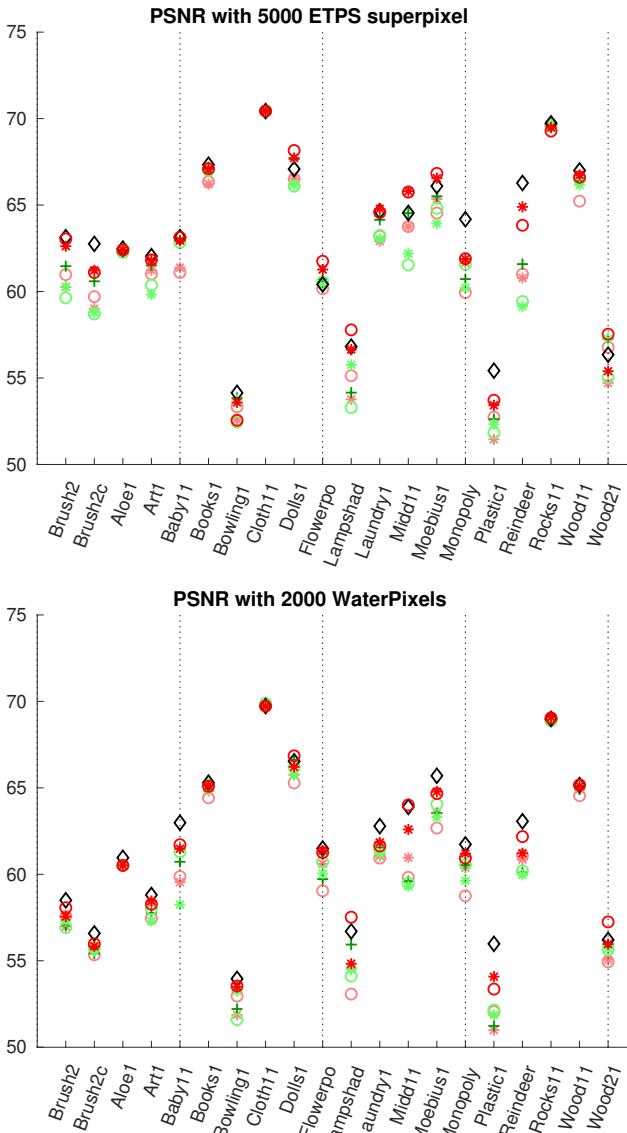


Figure C.5 – Per scene best results in terms of PSNR measure for each neighborhood construction using either 5000 ETPS superpixels (top) or 2000 WP (bottom). The higher the value is, the better the result is.

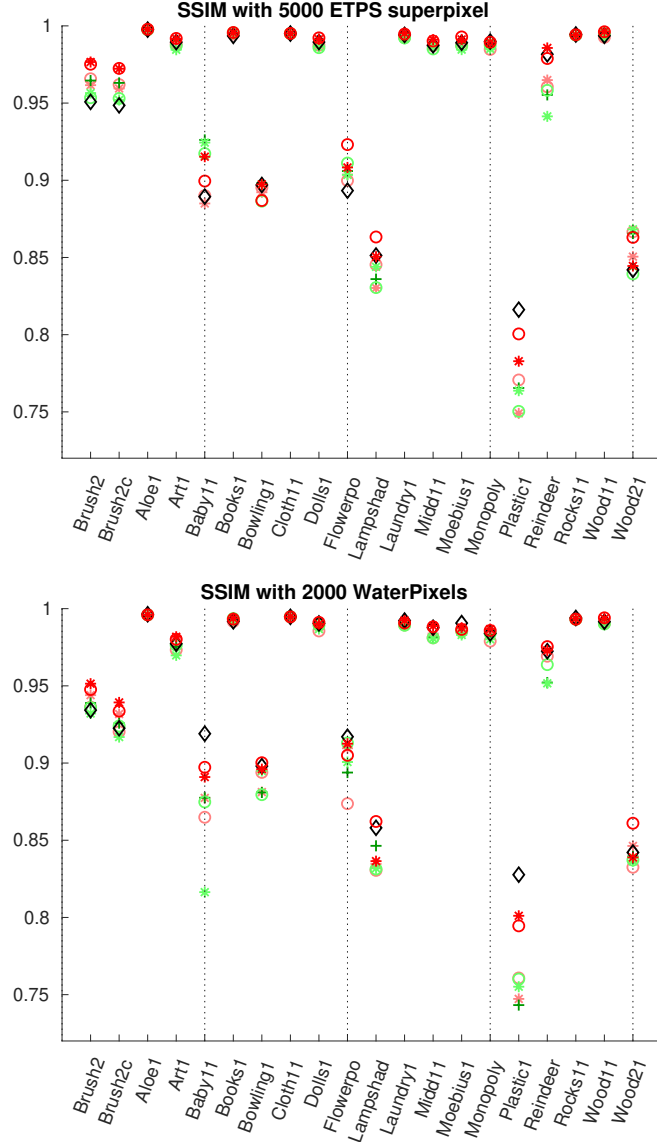


Figure C.6 – Per scene best results in terms of SSIM for each neighborhood construction; 5000 ETPS superpixels (top) or 2000 WP (low). The higher the value is, the better the result is.

We also note that lowest values are achieved for the perfect neighborhood (by construction) and then by CBN (either from TVo or RORPO guidance map) whereas both TBN and Stawiaski's neighborhood are much worse in terms of homogeneity.

Secondly, we compare the guidance maps provided by TVo and by RORPO. Figure C.8 shows these maps in the cases of the two considered scenes, *Lampshade* and *Reindeer*. In both cases, we notice that the direction of the structures is rather well estimated although we also observe some noise. Comparing the two estimators, we note that while TVo looks smoother in terms of orientation (especially on *Reindeer* scene), RORPO both better detects the isotropic areas (in white) and highlights well the sharp areas of the scene. However, these observed differ-

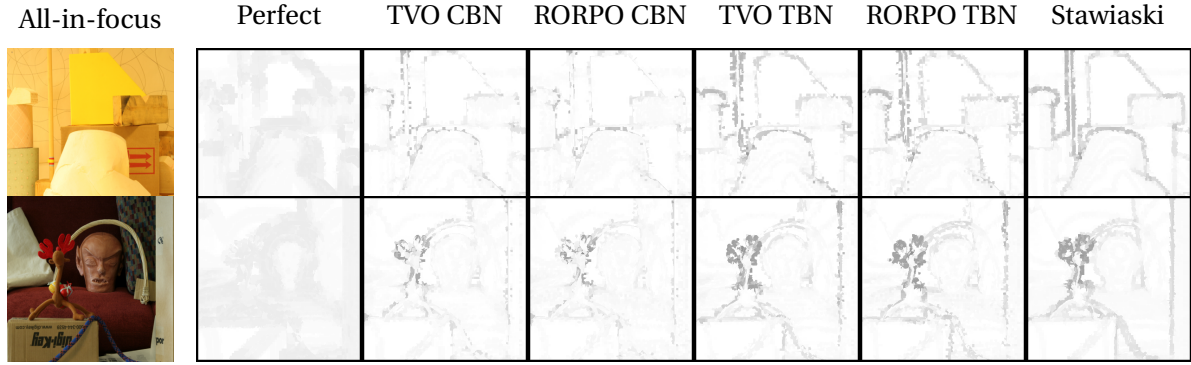


Figure C.7 – Comparison of neighborhood quality Q_V for *Lampshade* (top row) and *Reindeer* (bottom row) scenes, from left to right: All-In-Focus image, Q_V maps for *perfect* neighborhood, TVo CBN, RORPO CBN, TVo TBN, RORPO TBN and Stawiaski's neighborhood. Dynamics has been reversed and spread in the interval $[0, 255]$ so that dark areas represent bad performance.

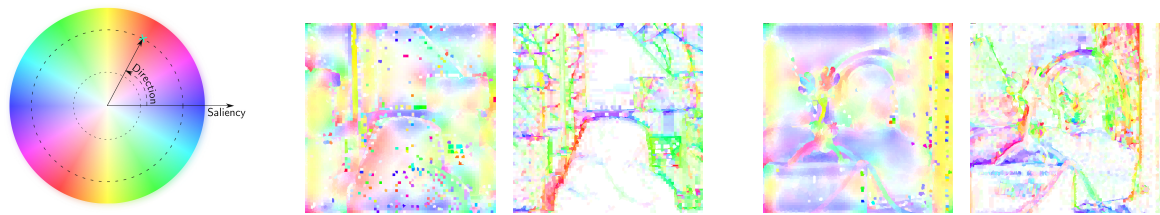


Figure C.8 – Comparison of guidance maps \mathbf{g} for *Lampshade* and *Reindeer* scenes, using a color representation, such that the saturation and the hue encode respectively the saliency and the orientation; from left to right: Color wheel, TVo *Lampshade*, RORPO *Lampshade*, TVo *Reindeer*, RORPO *Reindeer*.

ences seems to have only little impact on the neighborhood consistency as depicted in Figure C.7 or on depth map reconstruction. In what follows, we now focus on RORPO algorithm.

Finally, let us observe the error maps versus regularization parameter α for our two scenes and the three methods of neighborhood estimation: *Perfect* (reference for benefit of anisotropic neighborhood), RORPO CBN and Stawiaski (reference for isotropic neighborhood). For *Lampshade* scene, we notice the very high noise level in the absence of regularization ($\alpha = 0$) that is progressively corrected by increasing α before new errors this time due to the removal of thin structures appear. This phenomena can be clearly seen in the case of Stawiaski's neighborhood with apparition of errors located on the vertical thin bar or rod for $\alpha > 1$. From this scene, we also notice that the optimal α values vary with the considered neighborhood; as expected, anisotropic neighborhoods allow for higher α values without reconstruction degradation (in particular for the thin structures). Specifically, in *Lampshade* scene, α values providing best results are equal to 4, 8 and 2 for the *Perfect*, RORPO CBN and Stawiaski' neighborhoods, respectively. *Reindeer* scene is much less noisy than *Lampshade* scene. However, regularization is again

required to remove the *blind* estimation errors in the vertical right strip and in the bottom triangle, both been part or subparts of objects presenting a very homogeneous radiometry. Due to this lower initial level of noise, α "optimal" values are lower than in *Lampshade* scene, namely they are equal to 1, 2 and 0.5 for the *Perfect*, RORPO CBN and Stawiaski' neighborhoods, respectively. We notice that for higher values, regularization introduce depth errors on the antlers of the reindeer figure, all the more quickly as the neighborhood is isotropic (indeed with Stawiaski, bottom triangle errors cannot be corrected without degrading reindeer antlers). Using anisotropic neighborhoods, either *Perfect* or RORPO CBN, the degradation of thin structures is delayed so that we observe the existence of α values allowing for the correction of *blind* errors without introduction of new errors.

Superpixel versus pixel level

Finally, let us investigate the benefit of considering the superpixel level rather than the pixel one. For doing so, we consider again global performance statistics, namely the RMSE computed on the whole dataset.

In terms of complexity, the number of pixels is 360×360 versus 5000 superpixels (ETPS) in the con-

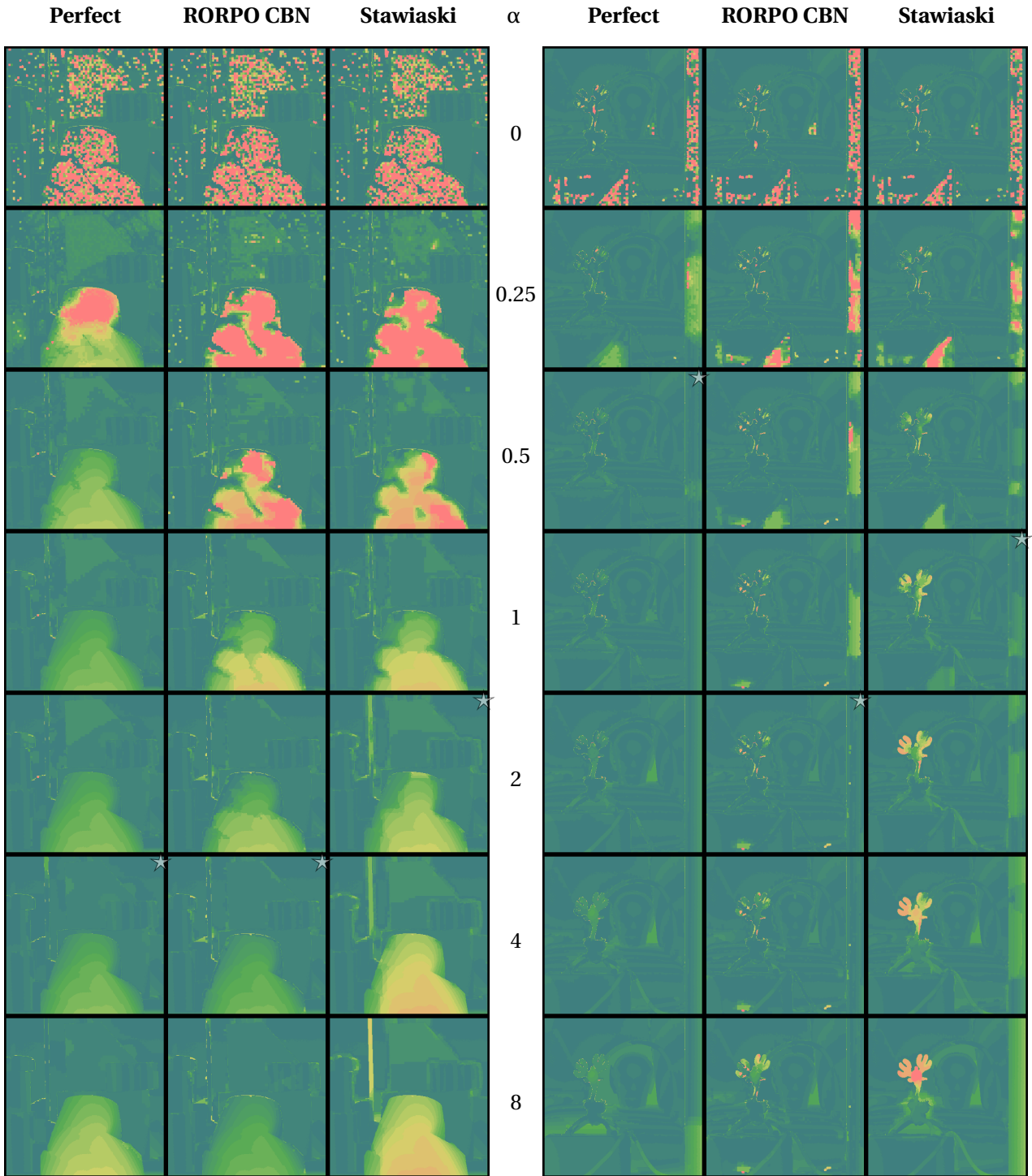


Figure C.9 – Maps of depth error obtained for the scenes *Lampshade* (half left) and *Reindeer* (half right), for three neighborhood construction strategies (*Perfect*, *RORPO CBN* and *Stawiaski*) and different values of regularization parameter $\alpha \in \{0, 0.25, 0.5, 1, 2, 4, 8\}$. For better visualization, error value dynamic has been bounded to $\frac{2\Delta_h}{5}$.

sidered experiments. Table C.1 gives the mean running times in seconds computed over all the scenes of the Middlebury college dataset, for the four main steps of our approach: (i) superpixel segmentation, (ii) guidance map estimation (either based on RORPO or on TVo), (iii) neighborhood construction, and (iv) depth map optimization. These running times have been obtained on an Intel core i9-10900X @ 4.7 GHz, with 64 Go of RAM. Table C.1 firstly confirms that RORPO is much faster (3 times) than TVo. Secondly, considering RORPO instead of TVo, the average running time for the global algorithm is 53.7 secs, i.e. about 1 minute per image. We consider this time as very encouraging since it was achieved with standard programming code, i.e. without optimization using GPU for instance. Thirdly, the running time for depth map optimization (using graph cuts) is very low thanks to the complexity reduction working with superpixels instead of pixels. For comparison, running the isotropic neighborhood depth map optimization at pixel level, the average running time is 36.8s, i.e. about 30 times slower. Thus, even without code optimization, the additional running time for anisotropic neighborhood estimation steps is compensated by the running time decrease for depth map optimization step.

| Superpixels segmentation | RORPO | TVo | CBN construction | Depth optim. |
|--------------------------|-------|------|------------------|--------------|
| 13.2 | 28.1 | 94.6 | 11.1 | 1.3 |

Table C.1 – Mean running times (in seconds) of the main steps of our approach for computing segmentation of a scene at superpixel level.

| | Lampshade | | Reindeer | |
|------------------|--------------|--------------|--------------|--------------|
| | PSNR | SSIM | PSNR | SSIM |
| RORPO-CBN ETPS | 57.78 | 86.33 | 63.83 | 97.89 |
| 4-adjacency pix. | <u>54.61</u> | <u>83.81</u> | <u>63.94</u> | <u>96.40</u> |
| RORPO-CBN pix. | 53.32 | 83.00 | 64.62 | <u>97.18</u> |

Table C.2 – Results obtained for the scenes *Reindeer* and *Lampshade* with our proposed anisotropic neighborhood at superpixel level (first row), compared to isotropic neighborhood at pixel level (second row) and RORPO-CBN neighborhood at pixel level (last row). SSIM values are indicated in percentage. For each scene, best result is in bold and second best is underlined.

In terms of performance, Figure C.10 allows for comparison of the RMSE curves for three kinds of neighborhoods, namely *Stawiaski* (i.e., isotropic), RORPO CBN or RORPO TBN (representing best can-

didate for anisotropic neighborhoods) and *Perfect*, either at superpixel level (using 5000 ETPS superpixels) or at pixel level. First of all, from Figure C.10, we notice an improvement of performance at superpixel level with respect to pixel one. This improvement is a very strong point since one could have expected that superpixels would introduce some spatial imprecision (at the benefit of complexity decrease), especially since the RMSE is measured at pixel level. Nevertheless, at least on the considered dataset, this preprocessing step is beneficial for the precise image reconstruction. This comment is confirmed in most cases when we examine individual scenes. For instance, for the two detailed cases *Lampshade* and *Reindeer*, the two first lines of Table C.2 show the performance indicators PSNR and SSIM achieved by RORPO-CBN on ETPS superpixels and isotropic (4-connectivity) and we see that RORPO-CBN yields to significantly better result except in terms of PSNR on *Reindeer* scene where nevertheless the performance values are very close.

Then, we notice the potential benefit of anisotropic neighborhood with respect to isotropic one (at pixel level, Stawiaski's neighborhood boils down to 4-connectivity neighborhood) since *Perfect* neighborhood yields the best results. However, we also notice that, at pixel level, the difference of performance is very small, and that isotropic neighborhood yields slightly better result than RORPO TBN or RORPO CBN. A possible explanation is that the requirement to take into account anisotropic neighborhood is less pregnant at pixel level (due to the size of neighborhood with respect to objects in pixel numbers as well as the regularity of the lattice) and that neighborhood estimation is less efficient. Indeed it is based on *blind* depth estimation that may be much noisier at pixel level than at superpixel one. Besides, the performance may depend on the considered scene. For instance, Table C.2 shows that on the *Reindeer* scene, RORPO-CBN at pixel level slightly outperforms isotropic pixel level both in terms of PSNR and SSIM indicators. These observations also open perspectives to understand the relationship between scene feature and scale of analysis (from pixel level to superpixel ones).

In conclusion, the benefit of presegmenting the scene in superpixels and then handling anisotropic neighborhood appears both in terms of global performance and in terms of robustness with respect to regularization parameter α . Besides the additional complexity introduced by neighborhood estimation (RORPO-CBN according to this study) is com-

pensated by the complexity decrease when handling much less superpixels than pixels.

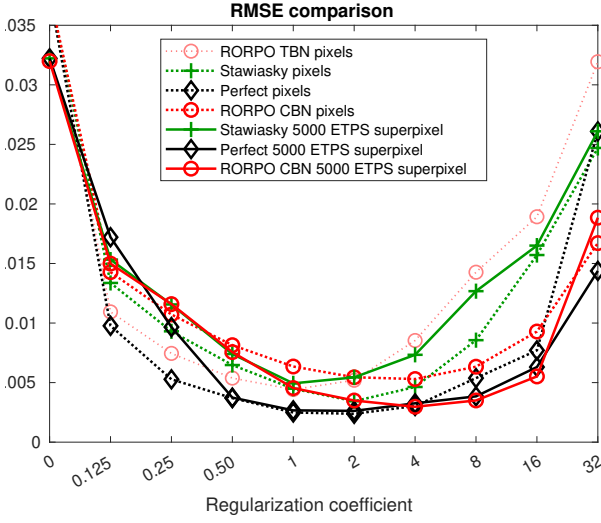


Figure C.10 – Superpixel versus pixel level: Comparison in terms of RMSE computed on the whole dataset, for three kinds of neighborhoods.

C.5 Conclusion and perspectives

In this paper, we propose some new anisotropic neighborhoods that offer a flexible and generic formulation with respect to the site lattice (i.e. possibly irregular). For doing so, we select and customize two vesselness operators and we show their efficiency thanks to their properties of noise robustness or adaptability to thin structures. Finally, we evaluate and study the benefit of the constructed anisotropic neighborhoods in particular for thin structure preservation. Specifically, we consider SFF application and we evaluate our results on a reference dataset both

according to quantitative criterion but also based on qualitative observation of evaluation maps.

Future works will involve the following perspectives. Firstly, we aim at studying the relationships between the hyperparameters characterizing the neighborhoods and the superpixel ones (regularity, number), also relating these parameters to the scale of scene main features and objects. Secondly, focusing on RORPO-CBN approach that appears to provide best performance and based on the evaluation of the running times per process, we will focus on the code optimization of the RORPO module. Thirdly, since the proposed anisotropic neighborhood construction can be useful for many energetic formulations of discrete inverse problem as confirmed by preliminary tests on binary segmentation [Ribal et al. \[2020\]](#), we aim at considering segmentation of thin structures such as frequently encountered in medical imaging (e.g., vessels) or remote sensing imagery (e.g., roads, rivers). Fourthly, in the proposed approach, neighborhood construction relies on guidance map itself estimated from a first (blind) evaluation of the solution. We aim at evaluating the benefit of using a current evaluation of the solution for our neighborhood construction process. Although such an alternate minimization seems attractive, first tests showed that the risk of divergence from the GT solution is real so that we will have to define rigorously the convergence conditions.

C.6 3D Tensor Voting

Note: This appendix has been removed from this copy of the article since it is essentially composed of the elements of Appendix [B.2.1](#).

Appendix D

References

- R. Achanta and S. Susstrunk. Superpixels and Polygons Using Simple Non-iterative Clustering. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 4895–4904, July 2017. ISBN 978-1-5386-0457-1. doi: 10.1109/CVPR.2017.520. URL <http://ieeexplore.ieee.org/document/8100003/>. 9
- R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süssstrunk. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, Nov. 2012. ISSN 0162-8828, 2160-9292. doi: 10.1109/TPAMI.2012.120. URL <http://ieeexplore.ieee.org/document/6205760/>. 61, 63
- M. Ahmad and Tae-Sun Choi. A heuristic approach for finding best focused shape. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(4):566–574, Apr. 2005. ISSN 1051-8215. doi: 10.1109/TCSVT.2005.844450. URL <http://ieeexplore.ieee.org/document/1413274/>. 80
- E. Aldea and S. Le Hégarat-Masclé. Robust crack detection for unmanned aerial vehicles inspection in an *a-contrario* decision framework. *J. Electron. Imaging*, 24(6), Dec. 2015. ISSN 1017-9909. doi: 10.1117/1.JEI.24.6.061119. URL <http://electronicimaging.spiedigitallibrary.org/article.aspx?doi=10.1117/1.JEI.24.6.061119>. 61, 72
- P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik. Contour Detection and Hierarchical Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, May 2011. ISSN 0162-8828, 2160-9292. doi: 10.1109/TPAMI.2010.161. URL <http://ieeexplore.ieee.org/document/5557884/>. 3, XV
- T. Asplund, A. Serna, B. Marcotegui, R. Strand, and C. L. Luengo Hendriks. Mathematical Morphology on Irregularly Sampled Data Applied to Segmentation of 3D Point Clouds of Urban Scenes. In B. Burgeth, A. Kleefeld, B. Naegel, N. Passat, and B. Perret, editors, *Mathematical Morphology and Its Applications to Signal and Image Processing*, volume 11564, pages 375–387. Springer International Publishing, Cham, 2019. ISBN 978-3-030-20866-0 978-3-030-20867-7. doi: 10.1007/978-3-030-20867-7_29. URL http://link.springer.com/10.1007/978-3-030-20867-7_29. Series Title: Lecture Notes in Computer Science. 10
- J. Besag. On the Statistical Analysis of Dirty Pictures. *Journal of the Royal Statistical Society: Series B (Methodological)*, 48(3):259–279, July 1986. ISSN 00359246. doi: 10.1111/j.2517-6161.1986.tb01412.x. URL <http://doi.wiley.com/10.1111/j.2517-6161.1986.tb01412.x>. 44
- Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In *International Conference on Computer Vision*, volume 1, pages 105–112. IEEE, 2001. ISBN 978-0-7695-1143-6. doi: 10.1109/ICCV.2001.937505. URL <http://ieeexplore.ieee.org/document/937505/>. 2, 16, XV

- Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, Sept. 2004. ISSN 0162-8828. doi: 10.1109/TPAMI.2004.60. URL <http://ieeexplore.ieee.org/document/1316848/>. 44, 50, 51, 53, XVIII
- Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, Nov. 2001. ISSN 01628828. doi: 10.1109/34.969114. URL <http://ieeexplore.ieee.org/document/969114/>. 44, 54, 56, 57
- B. Cui, X. Xie, X. Ma, G. Ren, and Y. Ma. Superpixel-Based Extended Random Walker for Hyperspectral Image Classification. *IEEE Transactions on Geoscience and Remote Sensing*, 56(6):1–11, 2018. ISSN 0196-2892, 1558-0644. doi: 10.1109/TGRS.2018.2796069. URL <http://ieeexplore.ieee.org/document/8291607/>. 2, XV
- D. Greig, B. T. Porteous, and A. Seheult. Exact Maximum A Posteriori Estimation for Binary Images. *Journal of the Royal Statistical Society, Series B (Methodological)*, 51(2):271–279, 1989. 44
- J. Darbon and M. Sigelle. Image Restoration with Discrete Constrained Total Variation Part I: Fast and Exact Optimization. *Journal of Mathematical Imaging and Vision*, 26(3):261–276, Dec. 2006. ISSN 0924-9907, 1573-7683. doi: 10.1007/s10851-006-8803-0. URL <http://link.springer.com/10.1007/s10851-006-8803-0>. 54, 57, 58
- X. Descombes, J.-F. Mangin, M. Sigelle, and E. Pechersky. Fine structures preserving markov model for image processing. In *Proceedings of the scandinavian conference on image analysis*, volume 1, pages 349–356. Citeseer, 1995. 15
- X. Descombes, F. Kruggel, and D. Von Cramon. Spatio-temporal fMRI analysis using Markov random fields. *IEEE Transactions on Medical Imaging*, 17(6):1028–1039, Dec. 1998. ISSN 02780062. doi: 10.1109/42.746636. URL <http://ieeexplore.ieee.org/document/746636/>. 5
- E. A. Dinic. Algorithm for solution of a problem of maximum flow in networks with power estimation. In *Soviet Math. Doklady*, volume 11, pages 1277–1280, 1970. 51
- J. Edmonds. Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems. page 17, 1972. 51
- P. Favaro. Recovering thin structures via nonlocal-means regularization with application to depth from defocus. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1133–1140. IEEE, June 2010. ISBN 978-1-4244-6984-0. doi: 10.1109/CVPR.2010.5540089. URL <http://ieeexplore.ieee.org/document/5540089/>. 3, XV
- P. F. Felzenszwalb and D. P. Huttenlocher. Efficient Graph-Based Image Segmentation. *International Journal of Computer Vision*, 59(2):167–181, Sept. 2004. ISSN 0920-5691. doi: 10.1023/B:VISI.0000022288.19776.77. URL <http://link.springer.com/10.1023/B:VISI.0000022288.19776.77>. 61, 63, 64
- P. F. Felzenszwalb and D. P. Huttenlocher. Efficient Belief Propagation for Early Vision. *Int J Comput Vision*, 70(1):41–54, Oct. 2006. ISSN 0920-5691, 1573-1405. doi: 10.1007/s11263-006-7899-4. URL <http://link.springer.com/10.1007/s11263-006-7899-4>. 44
- L. R. Ford and D. R. Fulkerson. Maximal Flow Through a Network. *Can. j. math.*, 8:399–404, 1956. ISSN 0008-414X, 1496-4279. doi: 10.4153/CJM-1956-045-5. URL https://www.cambridge.org/core/product/identifier/S0008414X00036890/type/journal_article. 46
- D. Freedman and P. Drineas. Energy Minimization via Graph Cuts: Settling What is Possible. page 8, 2005. 44, 47

- B. Fulkerson, A. Vedaldi, and S. Soatto. Class segmentation and object localization with superpixel neighborhoods. In *2009 IEEE 12th international conference on computer vision*, pages 670–677. IEEE, Sept. 2009. ISBN 978-1-4244-4420-5. doi: 10.1109/ICCV.2009.5459175. URL <http://ieeexplore.ieee.org/document/5459175/>. 2, XV
- V. Gaganov and A. Ignatenko. Robust Shape from Focus via Markov Random Fields. *Conference "GraphiCon'2009"*, pages 74–80, 2009. 13, 81, XVI
- R. Gauriau. *Shape-based approaches for fast multi-organ localization and segmentation in 3D medical images*. PhD thesis, Telecom ParisTech, 2015. 43
- S. Geman and D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, Nov. 1984. ISSN 0162-8828. doi: 10.1109/TPAMI.1984.4767596. URL <http://ieeexplore.ieee.org/document/4767596/>. 2, 15, I, XV
- R. Giraud, V.-T. Ta, A. Bugeau, P. Coupe, and N. Papadakis. SuperPatchMatch: An Algorithm for Robust Correspondences Using Superpixel Patches. *IEEE Transactions on Image Processing*, 26(8):4068–4078, Aug. 2017a. ISSN 1057-7149, 1941-0042. doi: 10.1109/TIP.2017.2708504. URL <http://ieeexplore.ieee.org/document/7935413/>. 3, 21, 37, 42, IV, XVI, XXIV
- R. Giraud, V.-T. Ta, and N. Papadakis. Evaluation framework of superpixel methods with a global regularity measure. *Journal of Electronic Imaging*, 26(6):061603, July 2017b. ISSN 1017-9909. doi: 10.1117/1.JEI.26.6.061603. URL <http://electronicimaging.spiedigitallibrary.org/article.aspx?doi=10.1117/1.JEI.26.6.061603>. II
- D. Goldfarb and W. Yin. Parametric Maximum Flow Algorithms for Fast Total Variation Minimization. *SIAM Journal on Scientific Computing*, 31(5):3712–3743, Jan. 2009. ISSN 1064-8275, 1095-7197. doi: 10.1137/070706318. URL <http://epubs.siam.org/doi/10.1137/070706318>. 8, 9
- S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *2009 IEEE 12th international conference on computer vision*, pages 1–8. IEEE, Sept. 2009. ISBN 978-1-4244-4420-5. doi: 10.1109/ICCV.2009.5459211. URL <http://ieeexplore.ieee.org/document/5459211/>. 3, XV
- H. P. Hariharan and T. Herfet. Light field compression by superpixel based filtering and pseudo-temporal reordering. pages 1–4. IEEE, Jan. 2018. ISBN 978-1-5386-3025-9. doi: 10.1109/ICCE.2018.8326153. URL <http://ieeexplore.ieee.org/document/8326153/>. 31
- H. Heijmans, M. Buckley, and H. Talbot. Path Openings and Closings. *J Math Imaging Vis*, 22(2-3):107–119, May 2005. ISSN 0924-9907, 1573-7683. doi: 10.1007/s10851-005-4885-3. URL <http://link.springer.com/10.1007/s10851-005-4885-3>. 8
- H. Ishikawa. Exact optimization for markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1333–1336, Oct. 2003. ISSN 0162-8828. doi: 10.1109/TPAMI.2003.1233908. URL <http://ieeexplore.ieee.org/document/1233908/>. 53
- O. Juan and Y. Boykov. Capacity Scaling for Graph Cuts in Vision. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, Rio de Janeiro, Brazil, 2007. IEEE. ISBN 978-1-4244-1630-1. doi: 10.1109/ICCV.2007.4408970. URL <http://ieeexplore.ieee.org/document/4408970/>. 51
- R. Kindermann and J. L. Snell. *Markov random fields and their applications*. Contemporary mathematics ; v. 1. American Mathematical Society, Providence, R.I, 1980. ISBN 978-0-8218-5001-5. 15, II

- V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, Feb. 2004. ISSN 0162-8828. doi: 10.1109/TPAMI.2004.1262177. URL <http://ieeexplore.ieee.org/document/1262177/>. 44, 47, 71
- M. Lachaize, S. Le Hégarat-Masclé, E. Aldea, A. Maitrot, and R. Reynaud. Evidential split-and-merge: Application to object-based image analysis. *International Journal of Approximate Reasoning*, 103:303–319, Dec. 2018. ISSN 0888613X. doi: 10.1016/j.ijar.2018.10.008. URL <https://linkinghub.elsevier.com/retrieve/pii/S0888613X18302986>. 6
- Y.-J. Liu, C.-C. Yu, M.-J. Yu, and Y. He. Manifold SLIC: A Fast Method to Compute Content-Sensitive Superpixels. pages 651–659. IEEE, June 2016. ISBN 978-1-4673-8851-1. doi: 10.1109/CVPR.2016.77. URL <http://ieeexplore.ieee.org/document/7780446/>. 2, XV
- O. Lézoray and L. Grady, editors. *Image Processing and Analysis with Graphs: Theory and Practice*. CRC Press, 1 edition, July 2017. ISBN 978-1-315-21728-4. doi: 10.1201/b12281. URL <https://www.taylorfrancis.com/books/9781439855089>. 8
- V. Machairas, M. Faessel, D. Cardenas-Pena, T. Chabardes, T. Walter, and E. Decenciere. Waterpixels. *IEEE Transactions on Image Processing*, 24(11):3707–3716, Nov. 2015. ISSN 1057-7149, 1941-0042. doi: 10.1109/TIP.2015.2451011. URL <http://ieeexplore.ieee.org/document/7140806/>. 62, 64, XXV
- M. T. Mahmood and I. H. Lee. Shape from focus based on 3D structure tensor using optical microscopy. *Microsc Res Tech*, 83(1):48–55, Jan. 2020. ISSN 1059-910X, 1097-0029. doi: 10.1002/jemt.23386. URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/jemt.23386>. 13
- H. Majidifard, P. Jin, Y. Adu-Gyamfi, and W. G. Buttler. Pavement Image Datasets: A New Benchmark Dataset to Classify and Densify Pavement Distresses. *Transportation Research Record*, 2674(2):328–339, Feb. 2020. ISSN 0361-1981, 2169-4052. doi: 10.1177/0361198120907283. URL <http://journals.sagepub.com/doi/10.1177/0361198120907283>. 70
- G. Medioni, C.-K. Tang, and M.-S. Lee. Tensor Voting: Theory and Applications. 2000. 3, 23, 26, 27, 29, 31, III, IV, XII, XIII, XVI, XVIII, XIX
- G. Medioni, P. Mordohai, and M. Nicolescu. The Tensor Voting Framework. In *Handbook of Geometric Computing*, pages 535–568. Springer-Verlag, Berlin/Heidelberg, 2005. ISBN 978-3-540-20595-1. doi: 10.1007/3-540-28247-5_16. URL http://link.springer.com/10.1007/3-540-28247-5_16. 28
- O. Merveille, H. Talbot, L. Najman, and N. Passat. Curvilinear Structure Analysis by Ranking the Orientation Responses of Path Operators. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(2):304–317, Feb. 2018. ISSN 0162-8828, 2160-9292. doi: 10.1109/TPAMI.2017.2672972. URL <http://ieeexplore.ieee.org/document/7862284/>. 3, 23, 32, 33, 35, 65, III, IV, XVI, XVIII, XIX, XX
- M. Moeller, M. Benning, C. Schonlieb, and D. Cremers. Variational Depth From Focus Reconstruction. *IEEE Transactions on Image Processing*, 24(12):5369–5378, Dec. 2015. ISSN 1057-7149, 1941-0042. doi: 10.1109/TIP.2015.2479469. URL <http://ieeexplore.ieee.org/document/7271087/>. 13, 79, 81, XVI
- H. Nair and C. Stewart. Robust focus ranging. pages 309–314. IEEE Computer Society Press, 1992. ISBN 978-0-8186-2855-9. doi: 10.1109/CVPR.1992.223258. URL <http://ieeexplore.ieee.org/document/223258/>. 13
- S. Nayar and Y. Nakagawa. Shape from focus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(8):824–831, Aug. 1994. ISSN 01628828. doi: 10.1109/34.308479. URL <http://ieeexplore.ieee.org/document/308479/>. 12, 13, 78, 79, XVI

- T. S. Nguyen, S. Begot, F. Duculty, and M. Avila. Free-form anisotropy: A new method for crack detection on pavement surface images. In *2011 18th IEEE International Conference on Image Processing*, pages 1069–1072, Brussels, Belgium, Sept. 2011. IEEE. ISBN 978-1-4577-1303-3 978-1-4577-1304-0 978-1-4577-1302-6. doi: 10.1109/ICIP.2011.6115610. URL <http://ieeexplore.ieee.org/document/6115610/>. 70
- E. Parzen. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, Sept. 1962. ISSN 0003-4851. doi: 10.1214/aoms/1177704472. URL <http://projecteuclid.org/euclid.aoms/1177704472>. 16
- S.-C. Pei, W.-W. Chang, and C.-T. Shen. Saliency detection using superpixel belief propagation. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 1135–1139. IEEE, Oct. 2014. ISBN 978-1-4799-5751-4. doi: 10.1109/ICIP.2014.7025226. URL <http://ieeexplore.ieee.org/document/7025226/>. 3, XV
- S. Pertuz, D. Puig, and M. A. Garcia. Analysis of focus measure operators for shape-from-focus. *Pattern Recognition*, 46(5):1415–1432, May 2013. ISSN 00313203. doi: 10.1016/j.patcog.2012.11.011. URL <http://linkinghub.elsevier.com/retrieve/pii/S0031320312004736>. 13, 79, 81, XVII, XXII
- Ren and Malik. Learning a classification model for segmentation. pages 10–17 vol.1. IEEE, 2003. ISBN 978-0-7695-1950-0. doi: 10.1109/ICCV.2003.1238308. URL <http://ieeexplore.ieee.org/document/1238308/>. 9
- C. Ribal, N. Lermé, and S. Le Hégarat-Masclé. Efficient graph cut optimization for shape from focus. *Journal of Visual Communication and Image Representation*, 55:529–539, Aug. 2018. ISSN 10473203. doi: 10.1016/j.jvcir.2018.06.029. URL <https://linkinghub.elsevier.com/retrieve/pii/S104732031830155X>. 2, 13, 54, 58, 79, 80, 81, XV, XVI, XVIII
- C. Ribal, N. Lermé, and S. Le Hégarat-Masclé. Thin Structures Segmentation Using Anisotropic Neighborhoods. In M.-J. Lesot, S. Vieira, M. Z. Reformat, J. P. Carvalho, A. Wilbik, B. Bouchon-Meunier, and R. R. Yager, editors, *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, volume 1237, pages 601–612. Springer International Publishing, Cham, 2020. ISBN 978-3-030-50145-7 978-3-030-50146-4. doi: 10.1007/978-3-030-50146-4_44. URL http://link.springer.com/10.1007/978-3-030-50146-4_44. Series Title: Communications in Computer and Information Science. XXX
- D. Scharstein and C. Pal. Learning Conditional Random Fields for Stereo. pages 1–8. IEEE, June 2007. ISBN 978-1-4244-1179-5 978-1-4244-1180-1. doi: 10.1109/CVPR.2007.383191. URL <http://ieeexplore.ieee.org/document/4270216/>. 81, XXII, XXIII
- D. Scharstein, R. Szeliski, and R. Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. pages 131–140. IEEE Comput. Soc, 2001. ISBN 978-0-7695-1327-0. doi: 10.1109/SMBV.2001.988771. URL <http://ieeexplore.ieee.org/document/988771/>. 13
- A. M. Schuck and D. P. Rosenbaum. *Promoting safe and health neighborhoods: What research tells us about intervention*. 2000. 8
- E. Spitzer. Markov Random Fields and Gibbs Ensembles. *The American Mathematical Monthly*, 78(2):142–154, Feb. 1971. ISSN 0002-9890, 1930-0972. doi: 10.1080/00029890.1971.11992710. URL <https://www.tandfonline.com/doi/full/10.1080/00029890.1971.11992710>. 14, II
- J. Stawiaski and E. Decenci re. Region merging via graph-cuts. *Image Analysis & Stereology*, 27(1): 39, May 2011. ISSN 1854-5165, 1580-3139. doi: 10.5566/ias.v27.p39-45. URL <https://www.ias-iss.org/ojs/IAS/article/view/828>. 2, 21, 42, 71, 84, 85, 94, 97, 98, 102, 115, 116, 119, 127, 129, II, III, IV, VI, XV, XXIV

- D. Stutz, A. Hermans, and B. Leibe. Superpixels: An evaluation of the state-of-the-art. *Computer Vision and Image Understanding*, 166:1–27, Jan. 2018. ISSN 10773142. doi: 10.1016/j.cviu.2017.03.007. URL <http://linkinghub.elsevier.com/retrieve/pii/S1077314217300589>. 2, 9, 10, 61, XV, XXII
- R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A Comparative Study of Energy Minimization Methods for Markov Random Fields with Smoothness-Based Priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(6):1068–1080, June 2008. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.70844. URL <http://ieeexplore.ieee.org/document/4420084/>. 2, 44, XV, XVIII
- N. H. Tran, D. Seo, D. Woo, Y. Won, and H. T. Tu. Alpha cut for interactive image segmentation of thin and elongated objects. *IET Image Processing*, 13(11):1951–1959, Sept. 2019. ISSN 1751-9667, 1751-9667. doi: 10.1049/iet-ipr.2018.5740. URL <https://onlinelibrary.wiley.com/doi/10.1049/iet-ipr.2018.5740>. 11
- M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool. SEEDS: Superpixels Extracted via Energy-Driven Sampling. In *Computer Vision – ECCV 2012*, volume 7578, pages 13–26. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-33785-7 978-3-642-33786-4. URL http://link.springer.com/10.1007/978-3-642-33786-4_2. 61, 63
- J. Vandoni, S. Le Hegarat-Masclé, and E. Aldea. Crack detection based on a Marked Point Process model. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 3933–3938. IEEE, Dec. 2016. ISBN 978-1-5090-4847-2. doi: 10.1109/ICPR.2016.7900249. URL <http://ieeexplore.ieee.org/document/7900249/>. 72
- M. Wang, X. Liu, Y. Gao, X. Ma, and N. Q. Soomro. Superpixel segmentation: A benchmark. *Signal Processing: Image Communication*, 56:28–39, Aug. 2017. ISSN 09235965. doi: 10.1016/j.image.2017.04.007. URL <http://linkinghub.elsevier.com/retrieve/pii/S0923596517300735>. 10, 61
- Z. Wang and H. R. Sheikh. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 13(4):14, 2004. 93, XXIII
- F. Y. Wu. The Potts model. *Reviews of Modern Physics*, 54(1):235–268, Jan. 1982. ISSN 0034-6861. doi: 10.1103/RevModPhys.54.235. URL <https://link.aps.org/doi/10.1103/RevModPhys.54.235>. 17, 71
- J. Yao, M. Boben, S. Fidler, and R. Urtasun. Real-time coarse-to-fine topologically preserving segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2947–2955. IEEE, June 2015. ISBN 978-1-4673-6964-0. doi: 10.1109/CVPR.2015.7298913. URL <http://ieeexplore.ieee.org/document/7298913/>. 10, 61, 63, 64, 71, 75, 97, XXIII, XXIV, XXV
- Y. Yu, H. Guan, and Z. Ji. Rotation-Invariant Object Detection in High-Resolution Satellite Imagery Using Superpixel-Based Deep Hough Forests. *IEEE Geoscience and Remote Sensing Letters*, 12(11):2183–2187, Nov. 2015. ISSN 1545-598X, 1558-0571. doi: 10.1109/LGRS.2015.2432135. URL <http://ieeexplore.ieee.org/document/7277008/>. 3, XVI
- Zhou Wang and A. Bovik. A universal image quality index. *IEEE Signal Processing Letters*, 9(3):81–84, Mar. 2002. ISSN 1070-9908, 1558-2361. doi: 10.1109/97.995823. URL <http://ieeexplore.ieee.org/document/995823/>. 93
- Q. Zou, Y. Cao, Q. Li, Q. Mao, and S. Wang. CrackTree: Automatic crack detection from pavement images. *Pattern Recognition Letters*, 33(3):227–238, Feb. 2012. ISSN 01678655. doi: 10.1016/j.patrec.2011.11.004. URL <https://linkinghub.elsevier.com/retrieve/pii/S0167865511003795>. 12, 61, 70, XIX

Appendix E

Acronyms list

BP An optimization algorithm. [44](#)

ETPS Extended Topology Preserving Segmentation, an algorithm of superpixel. [10](#), [61–65](#), [67](#), [71](#), [96](#), [97](#)

FF The maximum flow algorithm introduced by Ford and Fulkerson. [50](#), [51](#)

FH Felzenswalb and Huttenlocher algorithm of superpixels. [61–64](#)

FM An evaluation of performances based on the elements of the matrix confusion. [72](#), [75](#)

FN *False Negative*. [23](#), [71–73](#), [91](#)

FP *False Positive*. [23](#), [71](#), [72](#), [90](#), [91](#)

GC A key optimization algorithm for minimizing functions used in our implementation. [44](#)

ICM An optimization algorithm. [44](#)

IoU A method for computing similarities between two sets. [91](#)

MRF *Markov Random Field*. [2](#), [3](#), [14–17](#), [25](#), [47](#), [48](#)

PSF The function describing the response of an imaging system to a point source. [78](#), [79](#)

PSNR A quality measure. [87](#), [88](#), [93–95](#), [97](#), [99](#), [103](#), [104](#), [108](#), [109](#), [112](#), [113](#), [117](#), [118](#)

RMSE A quality measure. [93](#), [106](#), [116](#), [120–122](#), [127](#)

RORPO *Ranking the Orientation Responses of Path Operators*. [23](#), [32–37](#), [42](#), [60](#), [65](#), [67](#), [68](#), [83–88](#), [92](#), [94–106](#), [108–119](#), [127](#), [129](#), [130](#), [XXXIX](#)

SEEDS Superpixels Extracted via Energy-Driven Sampling, an algorithm of superpixel. [61–63](#)

SFF *Shape From Focus*. [3](#), [11–13](#), [27](#), [31](#), [32](#), [36](#), [54](#), [62](#), [65](#), [67](#), [68](#), [78–84](#), [86](#), [127](#), [XXXVII](#)

SLIC Simple Linear Iterative Clustering, an algorithm of superpixel. [61–63](#)

SMLAP Sharpness operator used in [SFF](#) computation.. [81](#), [98](#)

SSIM A quality measure. [93–95](#), [97](#)

TN *True Negative*. [91](#)

TP *True Positive*. 72, 90, 91

TRW An optimization algorithm. 44

TV *Tensor Voting*. 23, 26–28, 30, 32, 37, 41, 60, 65–67, 70, 72, 73, 75, 76, 83–88, 92, 94–105, 108–110, 112–114, 116–119, 127

TVA The total variation. 2, 23

UQI A quality measure. 93–95, 97

WP Waterpixels, an algorithm of superpixel. 61, 62, 64

Appendix F

Symbols list

A_V area of the shape in shape-based neighborhood. 36, 37, 39, 67

\mathcal{C} set of labels of the label field. 14, 16, 23, 25, 44, 53–60, 70, 71, 84, 93

\mathcal{D} a dictionary. 38

e ellipsis excentricity in shape-based neighborhood. 36, 37, 39, 67

\mathbb{F} feature space. 15, 16, 70, 81

F energy functional. 59, 60

Γ threshold used for selection between isotropic or anisotropic neighborhoods. 26, 36–39, 67, 115

\mathbf{g} guidance map. 23, 25, 31, 35–41, 84, IX

\mathfrak{t} size of the dictionary. 38, 67

\mathcal{N} cliques of the random field. 14, 17, 45, 47, 53, 55–57, 71, 82

N scene spatial dimation. 14, 16, 17, 20, 23, 25, 26, 31, 33

Π_δ set of paths (in RORPO computation). 39, 40

\mathcal{P} set of pixels of an image. 14, 15, 17, 20, 81, 84, 93

\mathcal{R} The set of connectedness relationships. 32, 33

\mathcal{S} a shape in shape-based neighborhood. 36–38

σ_T Tensor voting range. 26, 28–30, 60, 65, 67, 72, 73, XII

\mathcal{S} set of sites. 14, 16, 17, 20–23, 25, 31, 33–36, 38–40, 44, 45, 47–50, 53–62, 67, 70, 71, 82–84, 86, 90, 91, 106, XL

\mathcal{S}_3 set of sites (in 3D space). 17, 20, 21, 82–84

\mathbf{u} segmentation: a field of labels. 14–17, 23, 25, 44, 47, 49, 53–60, 70, 71, 82–84

\mathcal{V} adjacency relationship. 20, 21, 25, 32–34, 39

V neighborhood. 14, 16, 17, 20, 21, 23, 25, 26, 36, 37, 39, 40, 59, 60, 67, 68, 70, 71, 74, 82, 83, 90, 91, XXXIX, XL

\mathbb{V} set of existing neighborhood, $V: \mathcal{S} \mapsto 2^{\mathcal{S}}$. 14, 16, 20, 25, 36, 71, 82

W weighting function associated with the neighborhood. 17, 23, 68, 71

\rightsquigarrow connectedness relationship. 20, 33–35

Remerciements

Et voici. Une page qui se veut plus introspective que les autres, encore ce défi qui m'est si périlleux : comment répondre à une question, comment poser à l'écrit des mots qui, par quête d'exactitude, ou par pudeur, ou à l'inverse par peur d'en faire trop, ne trouveront jamais de forme définitive ? Réduire la taille des phrases. Mettre des points. Simplifier. Abréger. Pour faire simple tout en y laissant les formes, voici une approche géographique, qui débute au laboratoire SATIE, à Paris-Saclay. J'espère bien sûr, au cours du chemin, n'oublier personne, quand bien même se serait-on perdus en le parcourant.

Ma vie de thèse, c'est tout d'abord l'équipe ALCOFRIT, à commencer par Marie et Jennifer, qui m'ont accueilli avec enthousiasme, puis Andres, Hernan et Nicola dans le bureau d'en face, avec lesquels nous vivons les escapades sustentatoires et salvatrices du midi. Un petit groupe de doctorants, que Huiqin, Rémi et Alirezai rejoindront et contribueront à faire vivre. Tout ce beau monde gravite, de près ou de loin, autour de Sylvie, ma directrice de thèse, dont j'admire l'investissement et la disponibilité vis à vis de ses doctorants, et remercie les critiques et idées, toujours constructives. Tout aussi présent et disponible dans mon encadrement, merci à Nicolas, porteur du projet au sein de l'équipe MOSS.

Dans cette même équipe, je tenais à adresser des remerciements tout particuliers à Émi, pour sa grande sympathie et sa dévotion sans faille dans l'aide des jeunes doctorants face à leurs problèmes divers, et notamment informatiques. Dans ce dernier domaine, mes remerciements suivants vont tout naturellement Marius, et son solide cours de C++ : il m'a presque tout appris. Et enfin, je n'oublie pas Flavien, toujours prêt à prêter main forte pour résoudre les problèmes matériels, et ce dès les aurores, à peine débarqué du 91.06, provenance Massy Palaiseau, RER B.

De l'autre côté de ce RER justement, je tiens à remercier ceux qui ont fait mon bonheur au quotidien, dans le passé comme le présent. Je pense à mes nombreux colocataires à la Bikok, ceux qui m'ont accueilli et que j'ai la chance de croiser encore tantôt (Sophire, Lele, Junior, Pionon, Peau D'ours), et ceux qui m'ont entouré au cours de ces dernières années de thèse (Ferd, François, Chatan, Cheddar, Pépette, Nico, Daphné, Alice, Alice aussi, François aussi, Lala, Erredé, et nouvellement Edmée), une petite famille en soi. Je n'aurais pas pu finir sans vous tous.

Beaucoup de vécu, et de nombreuses rencontres au cours de ces années Cachannaises, et qui se poursuivent entre autres au *Hurlingo* grâce à la superbe bande qui y gravite. Une pensée particulière aussi pour les pas de danse de Jorel, les cocktails de Pierrick, la sauterelle de Chloé, les week-ends de Gaetan et Zoly bien sûr, et ceux à base de baileys et de trains ratés de Marge, Candy et Diane, les sorties de Jérémy, Mégane et les moulins troubles. Big up aux journées sauvées par Cat et ses précieux *Qt*-conseils, hommage aux collègues de prépa et du Coin² qui m'ont transformé, salutations aux compas du pic qui m'ont éveillé...

D'ailleurs, une pensée spéciale à ces trois qui sont là depuis quasi toujours, Thibaut, Jérémie, Flo. Peu de gens me connaissent depuis plus longtemps qu'eux, si ce n'est : la famille ! Caro, Papa, Maman, merci d'être là, tout simplement. Enfin, une pensée aussi à ceux qui nous ont quitté au cours de ce périple, et à ceux que je n'ai pas pu accompagner autant que je l'aurais souhaité en cette période.

Merci vous tous, et bien plus encore, pour ces interactions sociales protéiformes qui toutes à leur façon apportent sa vie à ce monde. À la prochaine !

Titre: Voisinages anisotropes de superpixels pour la segmentation d'images de structures fines

Mots clés: Traitement d'images – Techniques numériques ; Optimisation combinatoire ; Segmentation bayésienne ; Reconstruction d'image ; Vision par ordinateur ; Superpixels ; Structures fines ; Anisotropie ; Voisinage ; Régularisation.

Résumé: En vision robotique, segmenter une image consiste à décomposer cette image en régions homogènes, en associant un label à chaque élément la constituant. Cette thèse propose une approche générique vis à vis de ces éléments, pixels ou superpixels, en les désignant communément sous le terme de *sites*. La résolution du problème inverse qu'est la segmentation d'images est généralement rendue robuste au bruit grâce à la formulation d'une hypothèse Markovienne sur le champ des labels, et d'un a priori d'homogénéité des labels au sein des voisinages. Cependant, la solution optimale (ou régularisée) tend alors à présenter des artéfacts indésirables, notamment la perte prématurée des structures fines, définies comme des structures dont la taille est réduite selon au moins une dimension.

La construction de voisinages anisotropes adaptés à ces structures permet de pallier ce problème. Ces voisinages sont calculés après une première étape d'estimation des orientations des structures fines présentes dans l'image. Trois options, dont deux adaptées de la littérature, sont proposées pour réaliser cette

étape cruciale : la minimisation d'une énergie, le vote de tenseurs, et le RORPO.

À partir des cartes d'orientation obtenues, quatre méthodes de construction des voisinages anisotropes sont retenues. Tout d'abord, un voisinage défini par des formes géométriques est présenté, puis la restriction à un nombre fini de configurations pour chaque site permet de formuler un voisinage basé sur un dictionnaire. Enfin, deux voisinages basés sur des chemins entre sites (l'un à extrémités fixées et l'autre à taille constante) sont considérés, tous deux faisant intervenir une énergie à minimiser.

Dans ce manuscrit, les segmentations obtenues par estimateur du Maximum A Posteriori (obtenu à partir de coupes de graphes) avec les voisinages anisotropes proposés sont comparées à celles supposant un voisinage isotrope dans le cas de deux applications : la détection de structures fines et la reconstruction de cartes de profondeur en Shape From Focus. Les résultats des différentes variantes proposées sont évalués qualitativement et quantitativement dans le but de souligner les apports de la méthode proposée.

Title: Anisotropic neighborhoods of superpixels for thin structure segmentation

Keywords: Computer vision; Image processing–Digital techniques; Combinatorial optimization; Image reconstruction; Segmentation; Superpixels; Thin structures; Anisotropy; Neighborhood; Regularization

Abstract: In the field of computer vision, image segmentation aims at decomposing an image into homogeneous regions. While usually an image is composed of a regular lattice of pixels, this manuscript proposes through the term of *site* a generic approach able to consider either pixels or superpixels. Robustness to noise in this challenging inverse problem is achieved by formulating the labels as a Markov Random Field, and finding an optimal segmentation under the prior that labels should be homogeneous inside the neighborhood of a site. However, this regularization of the solution introduces unwanted artifacts, such as the early loss of thin structures, defined as structures whose size is small in at least one dimension.

Anisotropic neighborhood construction fitted to thin structures allows us to tackle the mentioned artifacts. Firstly, the orientations of the structures in the image are estimated from any of the three presented options: The minimization of an energy, Tensor Voting, and RORPO.

Secondly, four methods for constructing the actual neighborhood from the orientation maps are proposed: Shape-based neighborhood, computed from the relative positioning of the sites, dictionary-based neighborhood, derived from the discretization to a finite number of configurations of neighbors for each site, and two path-based neighborhoods, namely target-based neighborhood with fixed extremities, and cardinal-based neighborhood with fixed path lengths.

Finally, the results provided by the Maximum A Posteriori criterion (computed with graph cuts optimization) with these anisotropic neighborhoods are compared against isotropic ones on two applications: Thin structure detection and depth reconstruction in Shape From Focus. The different combinations of guidance map estimations and neighborhood constructions are illustrated and evaluated quantitatively and qualitatively in order to exhibit the benefits of the proposed approaches.