Déformation de courbes et surfaces multirésolution sous contraintes

Basile Sauvage

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

THÈSE

pour obtenir le grade de
DOCTEUR DE L'INPG

Spécialité : **« Mathématiques Appliquées »**

préparée au Laboratoire de Modélisation et Calcul (LMC-IMAG)
dans le cadre de **l'École Doctorale**
« Mathématiques, Sciences et Technologies de l'Information, Informatique »

présentée et soutenue publiquement par

Basile Sauvage

le 7 décembre 2005

Déformation de courbes et surfaces multirésolution sous contraintes

---

Directrice de thèse : Stefanie Hahmann
Co-directeur de thèse : Georges-Pierre Bonneau

JURY

Mme Gudrun ALBRECHT	Professeur - Université de Valenciennes	Rapporteur
M. Christophe SCHLICK	Professeur - Université de Bordeaux	Rapporteur
M. Roger MOHR	Professeur - INP Grenoble	Président
M. Dominique MICHELUCCI	Professeur - Université de Bourgogne	Examinateur
M. Gershon ELBER	Professeur - Technion, Israël	Examinateur
Mme Stefanie HAHMANN	Maître de Conférences - INP Grenoble	Directrice de thèse
M. Georges-Pierre BONNEAU	Professeur - Université Joseph Fourier	Co-directeur de thèse

Résumé

Déformation de courbes et surfaces multirésolution sous contraintes.

Dans le domaine de la modélisation géométrique comme dans le domaine de l'informatique graphique, les utilisateurs sont toujours en quête d'outils ergonomiques pour éditer et déformer des courbes et des surfaces. La construction de ces outils nécessite d'abord un choix pertinent de modèles mathématiques pour représenter ces objets géométriques. Ensuite, l'adjonction de contraintes géométriques, intégrées dans l'outil d'édition, peut faciliter la manipulation.

L'objet de ce manuscrit est d'étudier l'intégration de contraintes non linéaires dans la déformation multirésolution de courbes et de surfaces lisses. Nous abordons successivement la conservation de l'aire inscrite dans une courbe B-spline plane, la conservation du volume englobé par une surface B-spline, la conservation du volume englobé par une surface de topologie arbitraire (paramétrée sur un maillage triangulaire), et la conservation de la longueur d'une courbe linéaire par morceaux. Les modèles multirésolution, basés sur des analyses en ondelettes, permettent de créer aisément des déformations à différentes échelles sur des objets complexes, tout en conservant les détails fins. Les contraintes sont calculées dans la base multirésolution, puis intégrées grâce à des optimisations sous contraintes. Les déformations gagnent ainsi en réalisme, sans que l'utilisateur n'ait à intervenir. Les méthodes que nous développons fonctionnent interactivement, et sont étudiées pour s'adapter à différents types de déformations.

Mots-clés: analyse multirésolution, courbes, surfaces, déformation sous contraintes, aire, longueur, volume, modélisation géométrique, B-spline.

Abstract

Constrained deformation of multiresolution curves and surfaces.

Building intuitive tools for manipulating curves and surfaces is a challenging task in the domains of geometric modelling and computer graphics. First, such tools need an appropriate mathematical model of the geometric objects. In addition, geometric constraints can enhance ergonomics, once they have been integrated into the editing tool.

We investigate the integration of non linear constraints into the multiresolution deformation of smooth curves and surfaces : area preserving of closed B-spine curves, volume preserving of B-spline surfaces, volume preserving of surfaces with arbitrary topological type (based on triangular meshes), and length preserving of piecewise linear curves. Thanks to multiresolution schemes based on wavelets one can easily deform complex objects at any scale, while keeping fine details. The geometric constraints are computed using the multiresolution basis. Then they are integrated through an automated constrained optimization step, and they prove to efficiently enhance the realism of deformations. Our methods work in real-time, and can be adapted to a broad range of situations.

Keywords: multiresolution analysis, curves, surfaces, constrained deformation, area, length, volume, geometric modeling, B-spline.

Remerciements

En premier lieu je tiens à remercier chaleureusement Stefanie Hahmann et Georges-Pierre Bonneau, qui ont encadré ces travaux de thèse. Trois ans durant ils m'ont fourni avec justesse les clés pour comprendre et apprendre ce métier. Leur aide m'a permis de poser mes premiers jalons dans le monde de la recherche.

Je remercie Gershon Elber, avec qui j'ai eu la chance de collaborer. La confiance qu'il m'a accordée est une précieuse invitation à l'ouverture. Son implication et son efficacité sont irréprochables.

Je remercie Gudrun Albrecht et Christophe Schlick pour m'avoir fait l'honneur de rapporter ce manuscrit. Leurs remarques constructives m'encouragent. Merci également à Dominique Michelucci et à Roger Mohr pour leur participation à mon jury.

J'adresse un merci reconnaissant à Jean Lorieux et à Thomas Carcaud pour leur aide au développement. Un merci souriant à Alex, avec qui j'ai partagé bien plus qu'un bureau. Merci à tous ceux qui m'ont entouré au LMC, et en particulier à celles qui mettent tant d'huile pour éviter que nous ne prenions les rouages administratifs en grippe.

Merci Alex pour le printemps qui ronfle chez toi comme un R de Brassens.
Merci Aude pour tes oreilles anticycloniques, gardiennes de mes émotions perdues dans les brumes d'automne.
Merci Claire pour les terres d'été où tu m'aides à garder les pieds.
Merci à vous, tous les amis fidèles du jeu des mille soirs et week-ends, les fondus de montagne, les mordus de musique, et Simon & Patrick.
Merci à ma famille.
Merci maman pour m'avoir soutenu dans tous mes choix.

Table des matières

Table des figures

Table des notations

1 Introduction

$\psi^j = (\psi_i^j)_i^T$ Vecteur colonne des ondelettes 5
$\varphi^j = (\varphi_i^j)_i^T$ Vecteur colonne des fonctions d'échelle 5
A^j et B^j Matrices de décomposition 5
P^j et Q^j Matrices de reconstruction 5
V^j Espaces d'approximation emboîtés 5
W^j Espaces de détails 5

2 Aire et courbes planes

β Paramètre de contrôle de la fonction objectif 22
$\mathcal{A}$ Aire signée incluse dans la courbe $\mathbf{c}$ 16
$\mathbf{c}(t)$ Courbe paramétrique plane 14
$\mathbf{c}^j = (\mathbf{c}_i^j)_i^T$ Vecteur des points de contrôle 14
$\mathbf{d}^j = (\mathbf{d}_i^j)_i^T$ Vecteur des détails 14
$\mathcal{D}$ Distance quadratique à (X_0, Y_0) 21
$\mathcal{E}_B$, $\mathcal{E}$ Énergie de tension 21
g lagrangien 23
H Matrice creuse pour l'expression de $\mathcal{E}$ 22
$I(\Gamma, \Theta)$ et $I(\gamma_i, \theta_j)$ Formes fonctionnelles bilinéaires anti-symétriques 16
M^e Matrice d'aire au niveau e 17
$N = p2^n$ Nombre de points de contrôle, dimension de V^n 14
X^e et Y^e Vecteurs coordonnées de la décomposition 16

3.2 Volume et surfaces B-spline non uniformes

Δ Déformation, élément de $\tilde{V}$ 37, 47
δ_i et $\tilde{\delta}_i$ Coefficients définissant Δ 37
Θ Fonction de mesure du volume 39
Θ_{YZ} et $\tilde{\Theta}_{YZ}$ Vecteurs de sommes partielles dans Θ 41, 43
θ_{ijk}^u et θ_{ijk}^v Coefficients intervenant dans Θ 40
$\theta_{\mathbf{ijk}}$ et $\tilde{\theta}_{\mathbf{ijk}}$ Coefficients intervenant dans Θ 40, 43
$\mathcal{L}$ Ensemble des indices des coefficients libres dans la minimisation 50
$\tilde{m}_u$ et $\tilde{m}_v$ Nombres de points de contrôle dans $\tilde{V}$ 37
m_u et m_v Nombres de points de contrôle dans V 35
$\tilde{N}_{\mathbf{i}}$ B-splines définies sur $\tilde{\mathbf{u}} \times \tilde{\mathbf{v}}$ engendrant $\tilde{V}$ 37
$N_{\mathbf{i}}$ B-splines définies sur $\mathbf{u} \times \mathbf{v}$ engendrant V 35
$\mathbf{p_i}$ Points de contrôle de $\mathbb{R}^3$ 35
S Surface composite 35
S^q Patch, élément de V 35

$\tilde{\mathbf{u}}$ et $\tilde{\mathbf{v}}$ Sous-séquences de $\mathbf{u}$ et de $\mathbf{v}$ 37
$\mathbf{u}$ et $\mathbf{v}$ Séquences de nœuds 35
$\tilde{V} \subset V$ Espaces de surfaces B-splines 35, 37

3.3 Volume et surfaces B-spline uniformes

$\delta\mathbf{c}^e$ Correction 71
$\varphi^j_{\mathbf{i}}$ Fonctions d'échelle 66
ψ^{j-1}_0, ψ^{j-1}_1 et ψ^{j-1}_2 Ondelettes 67
Θ Fonction de mesure du volume 69
$\Theta^q(\mathbf{c})$ Impact de $\mathbf{c}$ sur le volume de S^q 73
$\theta^e_{\mathbf{ijk}}$ Coefficients intervenant dans Θ 69
$\theta^{e,u}_{ijk}$ et $\theta^{e,v}_{ijk}$ Coefficients intervenant dans Θ 69
$\mathbf{c}^j = (X^j, Y^j, Z^j)$ Coefficients multirésolution 68
$\mathbf{d}^j_0$, $\mathbf{d}^j_1$ et $\mathbf{d}^j_2$ Coefficients de détail dans $\mathbb{R}^3$ 67
$\mathbf{p}^j_{\mathbf{i}}$ Points de contrôle de $\mathbb{R}^3$ 67
S^q Patch 67
$\mathbf{u}$ et $\mathbf{v}$ Séquences de nœuds 66

4 Volume et surfaces triangulaires

β Coefficient dans la subdivision de Loop 89
$\mathbf{c}^j$ Vecteur des points de contrôle 85
$\mathbf{c}^j(\mathcal{M}^j)$ Maillage de contrôle au niveau j 85
$\mathbf{d}^j$ Vecteur des coefficients de détail 85
$\mathcal{M}^j$ Maillage topologique au niveau j 83
N Filtre de mise à l'échelle 87, 89
R_i, R_p Filtres de prédiction 87, 88
$S(t)$ Surface limite 84
U Filtre de mise à jour 87, 92
$\mathcal{V}^j$ Ensemble des sommets de $\mathcal{M}^j$ 83
$\mathcal{W}^j$ Ensemble des sommets insérés par la subdivision au niveau j 83

5 Longueur et courbes tridimensionnelles

β Paramètre de contrôle de la fonction objectif 120
Δ_X, Δ_Y et Δ_Z Matrices bandes issues des contraintes 121
b Membre de droite issu des contraintes 121
$\mathbf{c}(t)$ Courbe paramétrée 3D 111
$\mathbf{c}^j = (\mathbf{c}^j_i)^T_i$ Vecteur des coefficients d'échelle *i.e.* des points de contrôle 112
C^j Vecteur des coefficients de tous les niveaux dans la décomposition à l'échelle j 112
C^j_K Vecteur des coefficients définissant la courbe à l'état K, décomposée au niveau j 113
$\mathbf{d}^j = (\mathbf{d}^j_i)^T_i$ Vecteur des coefficients d'ondelette, *i.e.* des détails 112
$\mathcal{D}$ Distance quadratique à C_A 120
$\mathcal{E}_B$, $\mathcal{E}$ Énergie de tension 119
f_i Contrainte de longueur 113
$\tilde{f}_i$ Contrainte de longueur linéarisée 121
g et $\tilde{g}$ lagrangiens 120
H Matrice bande pour l'expression de $\mathcal{E}$ 120
X^j, Y^j et Z^j Composantes de C^j 112

Chapitre 1

Introduction

1.1 Problématique

Dans le domaine de l'animation comme dans celui de la modélisation géométrique, les utilisateurs ont toujours été en quête d'outils intuitifs pour modéliser ou déformer des objets géométriques comme les courbes ou les surfaces. La construction de tels outils passe en premier lieu par le choix adéquat d'une représentation mathématique pour ces objets. Lorsque les objets manipulés deviennent complexes, il arrive alors que l'utilisation de contraintes, intégrées de façon transparente dans les outils d'édition, soit une aide ergonomique précieuse.

L'analyse multirésolution a reçu une attention particulière ces dernières années dans de nombreux domaines de la modélisation géométrique, de l'informatique graphique et de la visualisation. Elle est un outil puissant pour représenter efficacement des fonctions avec différents niveaux de détail. Dans ce cadre, une fonction est représentée par une "tendance grossière" qui contient l'information basse résolution, couplée avec une série de coefficients de détails qui codent l'information haute fréquence et qui permettent la reconstruction exacte du signal original.

La déformation d'objets complexes avec une grande quantité de détails s'avère souvent difficile et coûteuse en calculs. Dans un cadre multirésolution, de tels objets peuvent être édités à une échelle donnée, ce qui peut être exploité de deux manières : premièrement, une édition à basse résolution suivie de la réintégration des détails modifie la forme globale de l'objet, mais les détails caractéristiques sont préservés. Deuxièmement, la modification de détails fins change le caractère de la courbe sans changer sa forme globale.

Des représentations multirésolution basées sur les ondelettes ont été développées pour les courbes paramétriques [CQ92, FS94, GBS99] et peuvent être généralisées aux surfaces produit tensoriel, aux surfaces de topologie arbitraire [LDW97], aux données sphériques [SS95] et volumiques [CMPS97]. D'autres types de modélisations multirésolution existent pour des données définies sur des maillages irréguliers [Bon98] et pour des maillages arbitraires [ZSS97, KCVS98, Hop96, VP04]. Toutes ces techniques témoignent des nombreux intérêts que présentent la modélisation multirésolution pour la géométrie, mais rares sont les travaux qui abordent la multirésolution sous contraintes, bien que des domaines tels que la CAD/CAM ou l'informatique graphique puissent tirer profit de tels outils.

Dans le domaine de l'animation, la conservation de quantités géométriques telles que l'aire, le volume ou la longueur au cours de déformations sont des principes traditionnels de réalisme [Las87]. Grâce à l'augmentation des capacités de calcul, la simulation par modèles physiques

[TPBF87, DDCB00] y est aujourd'hui largement répandue, en particulier pour la réalisation de films d'animation. Cette méthode assure le réalisme physique du comportement des objets, mais elle s'avère souvent coûteuse en temps de calcul. Pour cette raison, il peut être judicieux d'utiliser des contraintes géométriques plus rapides à traiter, afin d'imiter les lois physiques qui régissent le comportement de ces objets.

Parallèlement, en modélisation géométrique, la construction d'outils intuitifs d'édition de courbes ou de surfaces dans les systèmes de conception géométrique assistée par ordinateur est un sujet de recherche actif depuis plusieurs décennies. Il est classiquement utile de pouvoir spécifier des contraintes linéaires telles que la position, la tangence ou la normale en un point.

L'application de contraintes linéaires, combinée avec une minimisation d'énergie sur la courbe ou la surface [Fow92, CG91, TQ94] ont prouvé leur efficacité pour les modèles de sculpture et pour l'animation. D'autres travaux abordent la tâche souvent ardue de gérer des contraintes non linéaires, telles que la déformation des courbes de Bézier à longueur constante [PJF97], la prescription de longueur pour des courbes de Bézier rationnelles [RP96], la conservation de volume pour des solides [RSB96] ou des surfaces implicites [DG95], la conservation d'aire pour des courbes fermées [Elb00]. Mais aucun de ces travaux n'intègre les contraintes dans un outil d'édition multirésolution.

Ce n'est que récemment que des contraintes non linéaires ont été incorporées dans des modèles multirésolution. En particulier Elber [Elb01] a proposé une édition multirésolution de courbes fermées avec conservation de l'aire à l'intérieur de la courbe. Par ailleurs le problème de la déformation de surfaces à volume constant a été traité par Botsch et Kobbelt [BK03] sous une forme particulière : une représentation multirésolution spécifique est construite autour de l'encodage d'éléments de volume entre les différentes échelles, au lieu de coefficients dans une base donnée. Ces travaux ont pour principal défaut des temps de calcul très importants, dus à de lourdes étapes d'optimisation. Citons enfin les travaux de Larboulette et Cani [LC04] qui proposent de créer des plis sur un maillage par propagation d'un profil, lui même généré par une déformation de courbe à longueur constante.

Dans ce contexte, nous proposons d'étudier un certain nombre de contraintes non linéaires dans la déformation multirésolution de courbes et de surfaces.

Le chapitre 2 présente un processus de déformation de courbes planes fermées, avec conservation de l'aire enfermée à l'intérieur de la courbe. Il s'agit de courbes B-splines uniformes munies d'une base d'ondelettes.

Dans le chapitre 3, nous proposons de conserver le volume englobé par des surfaces B-splines composites. Ces surfaces sont composées de patchs produit tensoriel, chacun acceptant une décomposition multirésolution. Nous étudions séparément le cas des B-splines uniformes et non uniformes, puis nous comparons ces deux modèles.

Ensuite nous abordons les surfaces multirésolution de topologie quelconque, paramétrées sur un maillage triangulaire semi-régulier (chapitre 4). Nous proposons une méthode de calcul du volume englobé par cette surface, puis de conservation au cours d'un processus de déformation.

Le chapitre 5 présente une méthode de déformation de courbes 3D linéaires par morceaux à longueur constante. Un schéma multirésolution est utilisé pour contrôler la taille et l'apparence des déformations. Cette méthode est ensuite encapsulée dans un outil d'édition de maillages, afin de modéliser la déformation d'objets mous et inélastiques.

Enfin un bilan de ces travaux est présenté dans le chapitre de conclusion. Il réutilise les concepts généraux, que nous présentons dans la suite de cette introduction, pour apporter des éléments de comparaison entre les méthodes décrites dans les différents chapitres. Ce bilan est suivi de propositions d'axes de recherche futurs.

Ce manuscrit est agrémenté d'une annexe, qui regroupe les principales formules sur les fonctions B-splines non uniformes, les ondelettes B-splines uniformes périodiques, puis les ondelettes B-splines uniformes sur un intervalle.

Le présent chapitre regroupe la définition des principaux outils. Cela passe ne premier lieu par les B-splines (section 1.2). Dans la section 1.3 nous fixons les concepts généraux d'une analyse multirésolution. Enfin dans la section 1.4, nous présentons des outils pour situer et analyser l'intégration des contraintes dans un processus de déformation.

1.2 Courbes et surfaces B-splines

Courbes de Bézier.

Historiquement, les courbes (et surfaces) de Bézier marquent le début de la CFAO (Conception et Fabrication Assistée par Ordinateur). Elles furent inventées en 1960 par Pierre Bézier qui travaillait alors chez Renault sur l'usinage de surfaces. P.F. De Casteljau s'est intéressé à ces problèmes chez Citroën (à la même époque mais ses travaux ne seront rendus publics qu'en 1975), laissant son nom à un algorithme de construction géométrique des courbes de Bézier.

Une courbe de Bézier $\mathbf{b}(t)$ de degré d est une courbe polynomiale paramétrique, qui s'exprime explicitement en fonction des points de Bézier $\mathbf{b}_i \in \mathbb{R}^2$ ou $\mathbb{R}^3$ comme l'application

$$\begin{aligned} \mathbf{b} \ &: I \subset \mathbb{R} \to \mathbb{R}^2 \text{ ou } \mathbb{R}^3 \\ t \quad &\to \mathbf{b}(t) = \sum_{i=0}^{d} \mathbf{b}_i \ B_i^d(t) \ , \end{aligned}$$

à l'aide de la base de polynômes de Bernstein

$$B_i^d(t) = C_d^i \ t^i (1-t)^{d-i}, \quad i = 0, \ldots, d, \text{ et } t \in I = [0,1]$$

Le choix de l'intervalle $I = [0,1]$ n'est pas une restriction, car les courbes de Bézier sont invariantes par transformation affine du domaine de paramétrisation.

Cette courbe $\mathbf{b}(t)$ approxime la ligne polygonale de sommets $\mathbf{b}_i$, appelée polygone de contrôle, et interpole les deux extrémités $\mathbf{b}_0 = \mathbf{b}(0)$ et $\mathbf{b}_d = \mathbf{b}(1)$. Elle est de classe C^∞, se prête bien aux calculs de positions et de dérivées, et a un certain nombre de propriétés géométriques intéressantes, comme la propriété d'enveloppe convexe et l'invariance affine.

Un défaut de cette construction est que, pour augmenter le nombre de degrés de liberté, il faut augmenter le degré des polynômes B_i^d, ce qui pose des problèmes d'oscillations et de contrôle (chaque point influence toute la courbe). Ces problèmes sont résolus par les courbes de Bézier composites, aussi appelées courbes splines ou polynomiales par morceaux, qui consistent à mettre bout à bout des courbes de Bézier de faibles degrés. Le nouveau problème est la gestion du degré de continuité des raccords, qui impose des conditions linéaires entre les points de Bézier voisins. Les courbes B-splines apparaissent alors comme une façon commode de définir des courbes splines avec un certain degré de continuité aux raccords, grâce à l'utilisation d'une base appropriée (Basis-spline).

Courbes B-splines.

Une courbe B-spline de degré d est une courbe polynomiale par morceaux de degré d. Soit $\mathbf{t} = \{t_0, t_1, \ldots, t_{m+d}\}$ une suite croissante de points de $\mathbb{R}$, appelés *nœuds*. Les nœuds $\{t_0, t_1, \ldots, t_d\}$ et $\{t_m, \ldots, t_{m+d}\}$ sont appelés *nœuds de bord* alors que $\{t_{d+1}, \ldots, t_{m-1}\}$ sont appelés *nœuds intérieurs*.
On définit récursivement les fonctions de base B-spline de degré $k \leq d$ par :

$$N_i^k(t) = \frac{t - t_i}{t_{i+k} - t_i} N_i^{k-1}(t) + \frac{t_{i+k+1} - t}{t_{i+k+1} - t_{i+1}} N_{i+1}^{k-1}(t) \quad \text{pour } i = 0, \ldots, m + d - k - 1$$

avec

$$N_i^0(t) = \mathbf{1}_{[t_i, t_{i+1}]} \quad \text{pour } i = 0, \ldots, m + d - 1$$

Ces fonctions N_i^d ont nombre de propriétés intéressantes dont :
- Elles sont positives,
- Leur support $[t_i, t_{i+d+1}]$ est compact,
- Elles sont polynomiales de degré d sur chaque intervalle.
- Dans la mesure où les nœuds sont distincts, elles sont continues de classe C^{d-1} en chaque nœud.

L'espace engendré par la famille $(N_i^d(t))_i$ est l'ensemble des fonctions polynomiales de degré d par morceaux (sur les intervalles $[t_i, t_{i+1}]$, $i \in \{d, \ldots, m-1\}$), avec une continuité C^{d-1} en tous les nœuds intérieurs (à la condition qu'ils soient distincts). Une courbe B-spline $\mathbf{c}(t)$ se définit alors par ses coefficients (ou points de contrôle) $\mathbf{c}_i$ dans la base des N_i^d, comme l'application

$$\begin{aligned} \mathbf{c} \ &: [t_d, t_m] \to \mathbb{R}^2 \text{ ou } \mathbb{R}^3 \\ t \quad &\to \mathbf{c}(t) = \sum_{i=0}^{m-1} \mathbf{c}_i N_i^d(t). \end{aligned} \tag{1.1}$$

Cette écriture a plusieurs avantages en géométrie :
- les coefficients ont une influence locale (support compact des fonctions de base),
- la régularité est assurée par les fonctions de base (aucune contrainte sur les points),
- on connaît des algorithmes robustes pour calculer la valeur en un point, les dérivées, etc. [Far01],
- comme nous le verrons plus tard, il est possible de développer des schémas multirésolution à partir de ces courbes.

Les principales formules ainsi que les schémas multirésolution utilisant les B-splines sont détaillés en annexe A.

1.3 Analyse multirésolution

La multirésolution est une notion très utilisée depuis quelques années, dans des domaines variés des mathématiques appliquées et de l'informatique graphique. Il faut avant tout prendre garde au fait que le terme "multirésolution" endosse des acceptions assez différentes, plus ou moins précises mathématiquement, selon les applications auxquelles il se rapporte. Pour le moins, il signifie que l'objet étudié est analysé à plusieurs échelles (ou niveaux de résolution), tout en conservant une information spatiale (ou temporelle). Typiquement en traitement du signal, on

parle d'analyse spatio-fréquentielle, et c'est en cela qu'elle a constitué une avancée notable sur les analyses fréquentielles classiques de type Fourier. Comme tous les travaux présentés dans ce manuscrit sont basés sur des modèles multirésolution, il convient de préciser dès maintenant le sens que nous lui conférons.

L'*analyse multirésolution* est à comprendre au sens d'*analyse en ondelettes*, tel qu'il a été défini par Mallat [Mal89]. Il s'agit d'une théorie d'analyse du signal qui permet de combiner dans chacun des *coefficients* (formant une *décomposition* du signal) une information spatiale (pour nous, position géométrique et topologique) et une information fréquentielle (échelle d'influence du coefficient sur la courbe ou la surface). Depuis, les ondelettes ont trouvé des champs d'application aussi variés que l'analyse numérique [BCR91], le traitement du signal [Mer99], le traitement d'image [Mal89, CM01], la visualisation [GLDH97, BHN96] et l'informatique graphique [SDS96, GSCH93].

Nous présentons ici les concepts multirésolution généraux, en les illustrant par des exemples sur les courbes fermées. Pour plus de détails sur les aspects théoriques le lecteur se reportera à l'ouvrage de Mallat [Mal89], aux travaux de Finkelstein et Salesin [FS94] en ce qui concerne les courbes, ainsi qu'à l'ouvrage de Stollnitz et al. [SDS96].

Soit E un espace fonctionnel dans lequel se trouvent les objets manipulés (courbes ou surfaces) et soient $V^j \subset E$ des espaces d'approximation emboîtés, c'est-à-dire vérifiant $V^0 \subset V^1 \subset \cdots \subset V^n$. Dans notre cadre, ces espaces sont de dimension finie. Il existe pour chacun d'eux une base de *fonctions d'échelle* $(\varphi_i^j)_i$ à valeurs dans $\mathbb{R}$. Des *espaces de détails* W^j sont les complémentaires de V^j dans V^{j+1}, et possèdent une base d'*ondelettes* $(\psi_i^j)_i$. Pour la concision des notations φ^j et ψ^j désigneront les vecteurs colonne contenant ces fonctions. L'espace V^n se décompose alors :

$$V^n = V^{n-1} \oplus W^{n-1} = V^{n-2} \oplus W^{n-2} \oplus W^{n-1} = \cdots = V^0 \bigoplus_{j=0}^{n-1} W^j. \tag{1.2}$$

Cette décomposition de l'espace V^n est le fondement d'une analyse en ondelettes. Les objets que nous manipulerons (courbes, surfaces) sont des éléments de cet espace. Selon les besoins, ils seront décomposés sur les sous espaces V^j et W^j, en respectant les égalités (1.2). L'exposant j donne une information sur l'échelle (ou la fréquence) : j est grand pour les hautes fréquences (petite échelle) ; j est petit pour les basses fréquences (grande échelle). Autrement dit, on trouve dans l'espace W^j des détails (de la courbe ou de la surface) de taille plus ou moins petite selon que j est plus ou moins grand. Quant à eux, les espaces V^j contiennent une approximation de l'objet d'autant plus grossière (basse fréquence) que j est petit.

Les égalités (1.2) impliquent que les fonctions de base sont raffinables, c'est-à-dire qu'il existe, pour $j \in \{1, 2, \ldots, n\}$, des matrices P^j et Q^j telles que les équations suivantes sont vérifiées :

$$\begin{aligned} \varphi^{j-1} &= (P^j)^T \varphi^j \\ \psi^{j-1} &= (Q^j)^T \varphi^j \ . \end{aligned} \tag{1.3}$$

Symétriquement, il existe des matrices A^j et B^j qui permettent de reconstruire les fonctions d'échelle fine φ^j à partir des fonctions d'échelle grossière φ^{j-1} et des ondelettes ψ^{j-1} :

$$\varphi^j = (A^j)^T \varphi^{j-1} + (B^j)^T \psi^{j-1}. \tag{1.4}$$

Remarquons que $[P^j \mid Q^j]$ et $\begin{bmatrix} A^j \\ B^j \end{bmatrix}$ sont nécessairement des matrices carrées qui vérifient la condition de reconstruction

$$\begin{bmatrix} P^j & Q^j \end{bmatrix} \begin{bmatrix} A^j \\ B^j \end{bmatrix} = I. \tag{1.5}$$

Le choix des fonctions d'échelle et des ondelettes détermine la structure des matrices P^j, Q^j, A^j, et B^j . Pour la plupart des applications, il est préférable que ces matrices soient creuses, afin que le coût des changements de base soit allégé. C'est le cas dans tous nos travaux, car les fonctions de base sont à support compact.

Un signal de V^n, par exemple une courbe multirésolution $\mathbf{c}(t)$, peut s'écrire dans la base φ^n :

$$\mathbf{c}(t) = \sum_i \mathbf{c}_i^n \varphi_i^n(t) = (\mathbf{c}^n)^T(\varphi^n), \tag{1.6}$$

où $\mathbf{c}^n = (\mathbf{c}_i^n)_i^T$ est un vecteur colonne de *coefficients d'échelle*. Pour les courbes et les surfaces, ces coefficients $\mathbf{c}_i^n$ s'appellent également *points de contrôle*, qui sont les sommets du *polygone de contrôle*. Ils appartiennent à $\mathbb{R}^2$ ou $\mathbb{R}^3$ selon les cas.

Les relations (1.3) et (1.4) permettent de créer un signal basse résolution $(\mathbf{c}^{n-1})^T\varphi^{n-1}$, approximant $\mathbf{c}$, en utilisant le filtre passe-bas A^n :

$$\mathbf{c}^{n-1} = A^n\mathbf{c}^n.$$

Les détails perdus dans ce filtrage peuvent être récupérés dans un signal haute fréquence $(\mathbf{d}^{n-1})^T\psi^{n-1}$, en utilisant un filtre passe-haut B^n :

$$\mathbf{d}^{n-1} = B^n\mathbf{c}^n \ .$$

Ce processus de séparation d'un signal $\mathbf{c}^n$ en un signal grossier $\mathbf{c}^{n-1}$ et des détails $\mathbf{d}^{n-1}$ est appelé *décomposition* ou *analyse*. Les matrices A^n et B^n s'appellent *filtres de décomposition* ou *d'analyse*. La décomposition peut être répétée récursivement sur le nouveau signal $\mathbf{c}^{n-1}$. Finalement, le signal original sera décomposé en un signal basse résolution $\mathbf{c}^0$ et des détails $\mathbf{d}^0, \ldots, \mathbf{d}^{n-1}$ à chaque échelle. Ce processus récursif (voir FIG. 1.1) est connu sous le nom de *banc de filtres* [Mal89]. La figure 1.2 illustre ce processus : la courbe fine (à gauche), définie par 512 points de contrôle ($512 = dim(V^n) = dim(V^9)$), est approximée par des courbes de V^9 à V^2 (de gauche à droite). Sur la ligne du haut sont représentés les polygones de contrôle, qui définissent (dans la base de fonctions d'échelle) les courbes approximantes $(\mathbf{c}^e)^T\varphi^e \in V^e$, en correspondance sur la ligne du bas.

$$\begin{array}{ccccccccc} (\mathbf{c}^n) & \longrightarrow & (\mathbf{c}^{n-1}) & \longrightarrow & \cdots & & (\mathbf{c}^1) & \longrightarrow & (\mathbf{c}^0) \\ & \searrow & & \searrow & & & & \searrow & \\ & & (\mathbf{d}^{n-1}) & & (\mathbf{d}^{n-2}) & \cdots & & & (\mathbf{d}^0) \end{array}$$

FIG. 1.1 – Banc de filtres.

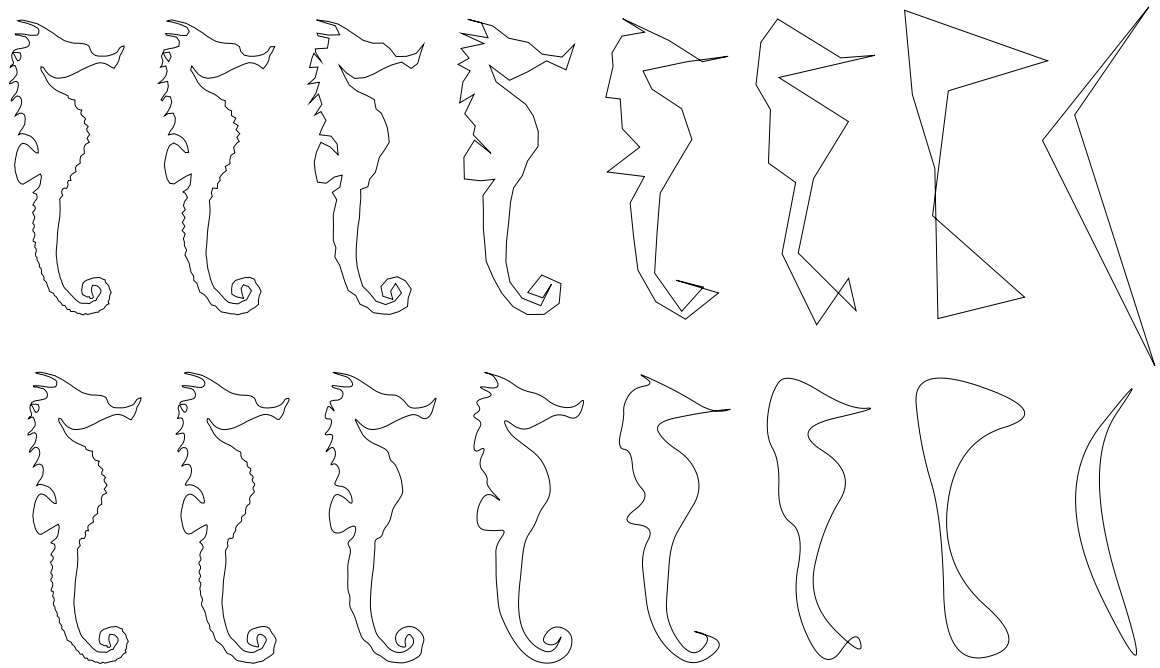


Fig. 1.2 – Décomposition.
Schéma d'ondelettes B-spline quadratiques. De gauche à droite : pour $e = 9, \ldots, 2$, polygones de contrôle $\mathbf{c}^e$ (en haut) et courbes approximantes $(\mathbf{c}^e)^T \varphi^e$ (en bas).

Par un processus appelé *reconstruction*, le signal original $\mathbf{c}^n$ est reconstitué à partir des $\mathbf{c}^0, \mathbf{d}^0, \ldots, \mathbf{d}^{n-1}$, en utilisant les *matrices de reconstruction* (ou *de synthèse*) P^j et Q^j :

$$\mathbf{c}^j = P^j \mathbf{c}^{j-1} + Q^j \mathbf{d}^{j-1} \qquad \text{pour } j = 1, \ldots, n.$$

Le filtre P^j est dit *filtre de subdivision*, car, d'un point de vue géométrique, il opère une subdivision du polygone de contrôle $\mathbf{c}^{j-1}$. Le filtre Q^j est dit *filtre de correction*, car il permet de corriger le polygone subdivisé $P^j \mathbf{c}^{j-1}$ en utilisant les détails $\mathbf{d}^{j-1}$ à l'échelle j, afin d'obtenir le polygone $\mathbf{c}^j$, plus fin, contenant plus d'information géométrique (échelles 0 à j).

En appliquant le banc de filtres partiellement, la courbe multirésolution $\mathbf{c}(t)$ peut être représentée à un certain *niveau de résolution* $e \in \{0, \ldots, n\}$ par des points de contrôle grossiers $\mathbf{c}^e$ qui forment une approximation du polygone de contrôle fin $\mathbf{c}^n$ et des coefficients de détail $\mathbf{d}^e, \ldots, \mathbf{d}^{n-1}$. On obtient :

$$\mathbf{c}(t) = (\mathbf{c}^e)^T(\varphi^e) + (\mathbf{d}^e)^T(\psi^e) + \ldots + (\mathbf{d}^{n-1})^T(\psi^{n-1}). \tag{1.7}$$

L'espace des paramètres des fonctions de base dépend fortement du choix de l'espace V^n, en ce sens que l'espace de paramétrisation contraint l'espace image V^n. Pour les courbes fermées du chapitre 2, il s'agit du tore unidimensionnel (ou $\mathbb{R}$ périodisé), et pour les courbes finies du chapitre 5 il s'agit d'un intervalle de $\mathbb{R}$. Pour les surfaces produit tensoriel (chapitre 3), il s'agit d'un pavé de $\mathbb{R}^2$, et pour les surfaces à base triangulaire (chapitre 4), il s'agit d'un maillage surfacique dans $\mathbb{R}^3$. Le choix de l'espace de paramétrisation détermine le type d'analyse en ondelettes, donc le type d'objets représentables et la forme des filtres de décomposition-reconstruction. Nous reviendrons sur ce sujet car il s'agit d'un premier élément de comparaison des méthodes

de déformations développées dans les différents chapitres.

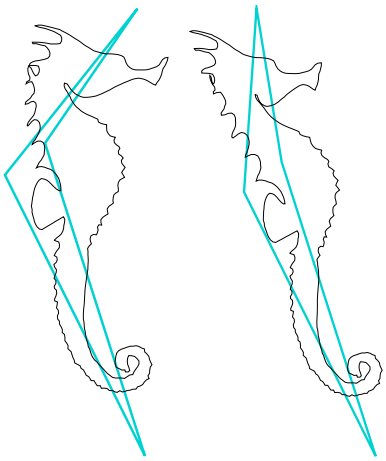


FIG. 1.3 – Édition des coefficients d'échelle.
La courbe initiale $\mathbf{c}(t) = (\mathbf{c}^9)^T \varphi^9(t)$ (à gauche) est manipulée par ses points de contrôle $\mathbf{c}^2$ (polygone gris), approximant la courbe à un niveau de résolution grossier $e = 2$.
Plusieurs de ces points sont déplacés (à droite) puis la courbe est reconstruite.
La forme globale est modifiée mais les détails sont inchangés.

Notre motivation à utiliser une représentation multirésolution est qu'elle sépare l'information par bandes de fréquences, entre les coefficients d'échelle et les coefficients d'ondelettes. Géométriquement, cela signifie que les détails fins seront codés dans $\mathbf{d}^j$ avec j proche de n, les détails grossiers dans $\mathbf{d}^j$ avec j proche de e, tandis que $\mathbf{c}^e$ encode toujours la forme globale. On ne perd pas pour autant l'information spatiale : le support compact des fonctions de base fait qu'un coefficient n'a qu'une influence locale. L'étendue de l'influence des coefficients croît quand le niveau j décroît. Il est donc plus aisé de manipuler un objet complexe. Appuyons nous sur l'exemple de la figure 1.3 : une courbe fine est décomposée de façon à obtenir 4 points de contrôle (à gauche). Ensuite des points de contrôle sont déplacés, puis les détails sont réintégrés (à droite). Ainsi la modification de coefficients d'échelle à un niveau bas change la forme globalement, mais la conservation de tous les détails fait que la courbe finale ressemble toujours à un hippocampe. Réciproquement, il est possible de modifier les coefficients de détails sans perturber la forme globale. Typiquement si on met les détails à 0 (FIG. 1.2) on obtient une approximation de la forme (courbes du bas).

1.4 Contraintes

Comment élaborer et analyser un modèle d'intégration de contraintes dans un processus de déformation ? Afin de situer et de comparer nos travaux sur la déformation sous contraintes, nous tâchons ici de placer les balises permettant de mener à bien cette lecture critique. L'élaboration d'un modèle de déformation contrainte est illustrée par le schéma 1.4.

Il faut tout d'abord déterminer *quelles contraintes* et quels *modèles de courbes et de surfaces* nous souhaitons traiter, en éclairant ces choix par les domaines d'application visés. Ensuite il est nécessaire de préciser les *méthodes de calcul* des contraintes, dépendantes des objets auxquelles

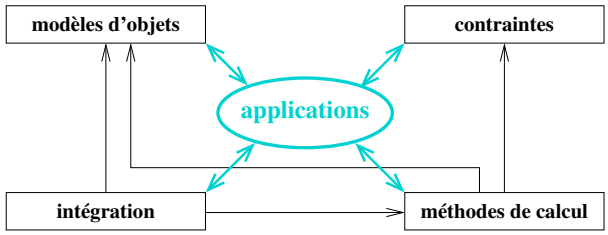


FIG. 1.4 – Élaboration d'un modèle de déformation contrainte.
Les applications assurent la cohérence entre les différents choix (boites) qui dépendent les uns des autres (flèches).

les contraintes s'appliquent. Ceci étant fait, il reste à définir la *manière* dont vont être *intégrées* les contraintes dans le processus de déformation. Les *applications*, effectives ou potentielles, sont en même temps une conséquence de tous ces choix, et un témoin de cohérence entre les choix.

Prenons pour exemple ce que nous développons au chapitre 2. L'objectif est d'apporter du réalisme (choix de la contrainte d'aire) dans l'animation d'objets déformables (choix de courbes B-splines). Nous mettons alors en place une méthode de calcul de l'aire sous forme multirésolution. Enfin, pour garder une méthode assez générale, nous choisissons l'intégration par une optimisation sous contraintes. L'application en animation est restreinte par ces choix en même temps qu'elle les justifie.

Le présent manuscrit est structuré à partir de ce schéma. Chacun des chapitres 2, 3, 4 et 5 concerne l'association d'une contrainte avec un modèle de courbe ou de surface. Chaque chapitre est décomposé en plusieurs sections consacrées : au modèle, au mode de calcul de la contrainte, puis à son intégration (déformation sous contrainte).

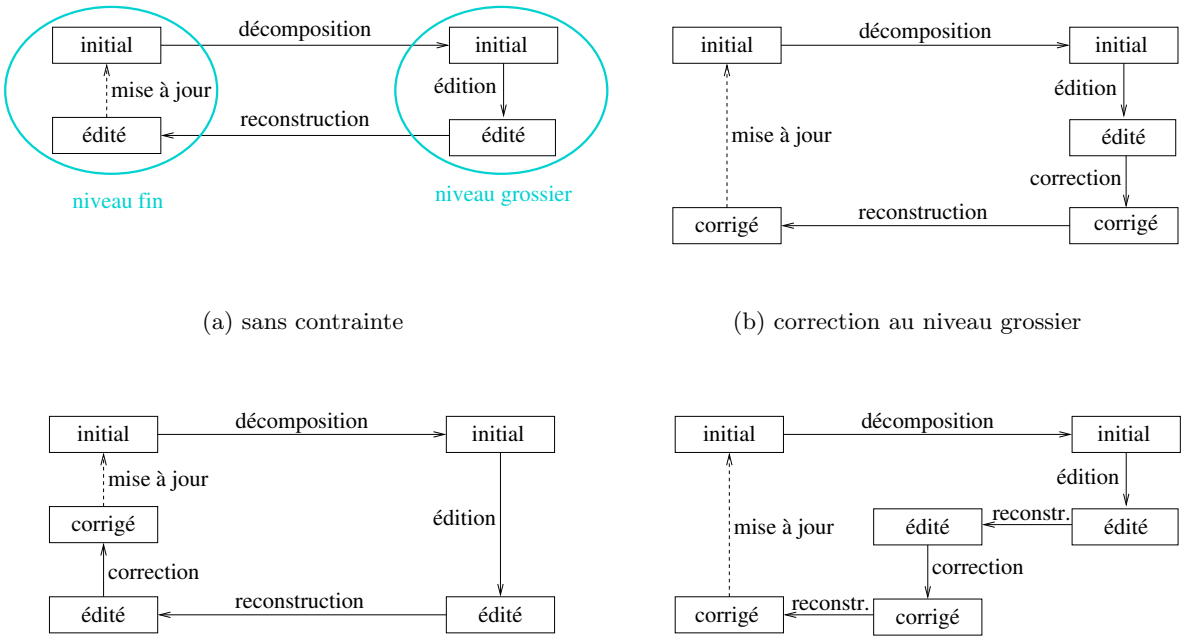
Intégration au processus de déformation.

Une fois définie la méthode de calcul d'une contrainte, il y a plusieurs manières de l'intégrer au processus de déformation. Nous présentons ici la notion de *boucle d'édition*, qui constitue un nouvel outil de comparaison des méthodes détaillées dans les différents chapitres.

L'objet que l'on souhaite déformer en vérifiant certaines contraintes peut être manipulé directement via un éditeur, via des contraintes extérieures (interaction avec d'autres objets d'une scène complexe), ou encore par un modèle d'animation quelconque. La cause possible de la déformation n'est pas notre sujet. Nous vérifions simplement que la réponse des contraintes est générique à la cause de déformation. Par conséquent, nous prenons la déformation comme un acquis, suite auquel nous allons modifier l'objet pour qu'il vérifie les contraintes. Nous avons néanmoins besoin de tester les méthodes, et nous utilisons pour cela des interfaces d'édition multirésolution développées par nos soins.

Une opération de déformation type est le "drag and drop" : un point de l'objet (ou un point de contrôle) est sélectionné à la souris puis déplacé par l'utilisateur. Chaque pas de déplacement de la souris déclenche l'exécution d'une "boucle d'édition" : le déplacement de souris définit une déformation de l'objet (appelée *édition* FIG. 1.5) à laquelle il faut répondre par l'application de la contrainte (appelée *correction* FIG. 1.5).

Prenons exemple sur la figure FIG. 1.6 qui illustre les boucles FIG. 1.5 (a) et (b). La courbe initiale fine (en haut à gauche) est *décomposée* à un niveau de résolution plus bas, aboutissant à un polygone de contrôle (en haut à droite) que l'on va *éditer* (étirement du nez du hérisson au milieu à droite). Sans appliquer de contrainte (FIG. 1.5(a)) on *recompose* alors la courbe fine



(a) sans contrainte

(b) correction au niveau grossier

(c) correction au niveau fin

(d) correction à un niveau intermédiaire

FIG. 1.5 – Boucles d'édition.
Les boîtes représentent les états de l'objet au cours du processus de déformation. Les transitions horizontales représentent des changements du niveau de résolution (décomposition ou reconstruction). Les transitions verticales représentent une modification de la géométrie de l'objet.

(au milieu à gauche) en réintégrant les détails pour obtenir la courbe finale qui est la courbe initiale pour la prochaine déformation. Cette courbe ne vérifie généralement pas les contraintes que l'on s'est fixées (l'aire constante dans FIG. 1.6). Suivant le modèle FIG. 1.5(b) il convient donc d'abord de *corriger* la courbe (en bas à droite) pour qu'elle vérifie la contrainte avant de la *reconstruire* (en bas à gauche).

Jusque là toutes les modifications géométriques (édition et correction) s'effectuent sur la courbe décomposée. Il ne semble donc pas nécessaire de recomposer la courbe à chaque boucle : cela sert uniquement à visualiser la courbe fine, donc à observer une réponse interactive à la déformation au niveau grossier. Cette remarque présuppose cependant que la correction est effectuée sur la décomposition de la courbe. Il peut arriver, pour des raisons que nous développerons en temps voulu, que la correction ait lieu à l'échelle fine (FIG. 1.5 (c)) ou à une échelle intermédiaire (FIG. 1.5 (d)). Nous envisagerons même de faire plusieurs corrections successives à des échelles différentes (chapitre 5). Dans ces cas-là, le jeu de décomposition-reconstruction est nécessaire.

Nous pourrons analyser et comparer nos méthodes, selon qu'elles observent un schémas ou l'autre. En effet, une correction au niveau grossier a l'avantage de ne pas imposer de reconstruction, et elle permet également de ne corriger que certaines bandes de fréquences du signal. Cependant, elle nécessite que la contrainte soit exprimée dans la base décomposée, ce qui n'est pas le cas pour une correction au niveau le plus fin (FIG. 1.5 (c)). Une correction à niveau

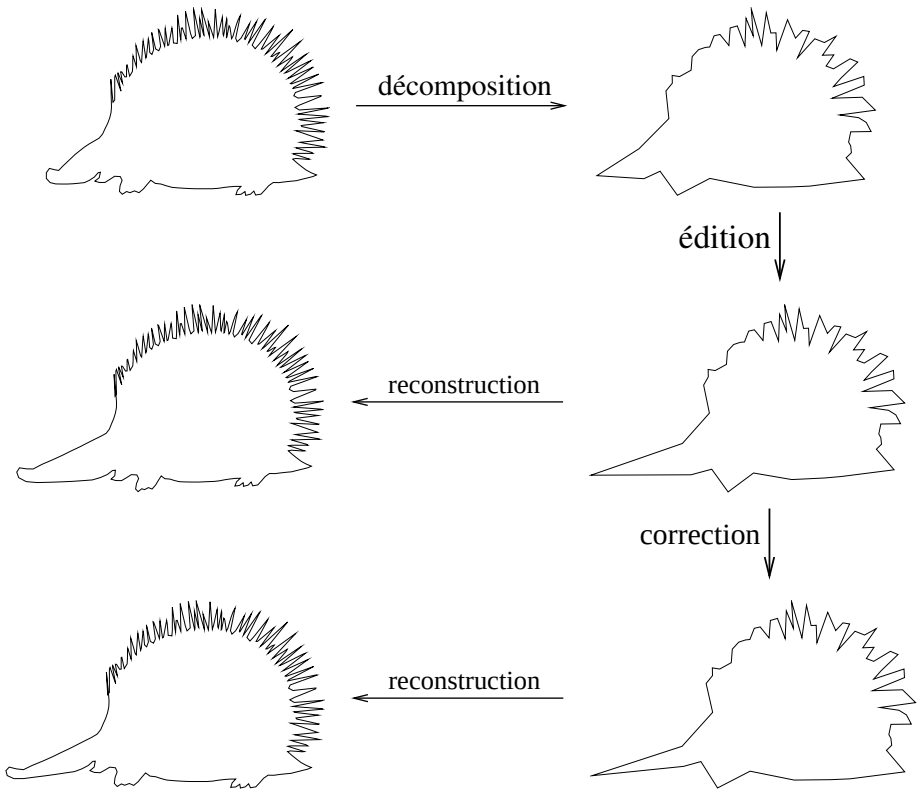


Fig. 1.6 – Édition et correction.
La courbe initiale (en haut) est éditée (au milieu) puis corrigée (en bas) pour maintenir l'aire constante. $n = 9$ pour les courbes fines (à gauche) et $e = 6$ pour les polygones de contrôle (à droite).

intermédiaire (Fig. 1.5(c)) offre une liberté supplémentaire (voir chapitre 5) : le choix de ce niveau est un paramètre de contrôle supplémentaire qui peut avoir une signification géométrique directe.

Aspects algorithmiques.

En développant les différentes méthodes, nous nous soucions de leur efficacité selon les deux critères classiques et souvent antagonistes : la complexité mémoire et la complexité en temps.

La *place mémoire* occupée lors des calculs devient vite importante avec les surfaces. Elle est donc souvent déterminante quand il s'agit de savoir quelle *taille de données* on peut traiter. La *complexité en temps* des algorithmes, autant que la qualité de leur implémentation, autorise ou non des déformations en *temps réel*, contraignant les possibilités applicatives.

Ces complexités, liées aux méthodes de calcul et d'intégration des contraintes, sont un fil rouge supplémentaire. Certains algorithmes et quelques structures de données sont détaillés, quand ils jouent un rôle primordial dans l'efficacité de la méthode.

Tous les résultats exposés dans ce manuscrit sont issus d'éditeurs codés par nos soins en `C++`, avec manipulation graphique interactive pour les chapitres 2 et 4, et manipulation par fichiers

de commandes pour les chapitres 3 et 5. Les interfaces graphiques utilisent les librairies `OpenGL` et `GLUI`. Les systèmes creux sont construits avec `SparseLib` et résolus avec `Iml++` (Iterative Methods Library). Les structures de données, l'analyse et la synthèse multirésolution ont été codées *ex nihilo*.

Chapitre 2

Aire et courbes planes

Nous étudions dans ce chapitre la contrainte d'aire pour les courbes fermées (périodiques). Nous décrivons une méthode de déformation multirésolution de courbes planes qui permet d'assurer la conservation de l'aire délimitée par la courbe. Les courbes sont exprimées dans une base d'ondelettes B-spline périodiques uniformes, et manipulées directement par leurs points de contrôle. La contribution principale de ces travaux est de fournir une méthode de calcul efficace de l'aire à partir de la décomposition en ondelettes. Nous proposons en outre un algorithme de déformation basé sur une linéarisation de la contrainte d'aire et sur une optimisation sous contrainte.

2.1 Problématique

La conservation de l'aire incluse dans une courbe fermée, lorsque celle-ci est déformée, est un élément de réalisme pour l'animation d'objets déformables. La contrainte d'aire a fait l'objet d'investigations pour les courbes B-splines non uniformes par Elber [Elb00], qui a ensuite étendu ses résultats [Elb01] pour appliquer des déformations à une échelle quelconque.

Nous proposons ici une alternative utilisant un schéma d'ondelettes B-splines uniformes périodiques. Ce cadre multirésolution permet de manipuler aisément la courbe par ses points de contrôle à n'importe quel niveau de résolution. Nous incorporons à ce processus d'édition la possibilité de conserver l'aire incluse dans la courbe grâce à une minimisation d'énergie sous contrainte. Via une minimisation partielle il sera en outre possible de contrôler l'étendue et l'aspect de la déformation. Pour construire un tel processus de déformation interactif, nous développons une méthode efficace et robuste de calcul d'aire dans la base multirésolution.

Ce chapitre est organisé comme suit. Dans la section 2.2, nous présentons le modèle de courbes utilisé. Ensuite, en section 2.3, nous expliquons comment calculer l'aire incluse dans de telles courbes, en utilisant le schéma multirésolution. Dans la section 2.4, nous développons une méthode d'intégration de la contrainte ainsi calculée dans un processus d'édition multirésolution. On linéarise la contrainte pour l'inclure dans la minimisation. Enfin, dans la section 2.5, on résume la méthode présentée dans [Elb01] et on la compare en détail avec notre méthode.

2.2 B-splines périodiques

Dans ce chapitre nous travaillons sur des courbes fermées (périodiques) planes, dans le cadre multirésolution général présenté en section 1.3. Nous présentons ici les points importants pour construire une analyse multirésolution dans cet espace, mais de plus amples détails sont donnés en annexe A.2.

L'espace V^n est de dimension $p2^n$ avec $p \in \mathbb{N}^*$, ce qui revient à dire qu'une courbe de cet espace est représentée au niveau fin par $N = p2^n$ coefficients $\{\mathbf{c}_0^n, \ldots, \mathbf{c}_{p2^n-1}^n\} \in \mathbb{R}^2$. Les espaces V^j pour $0 \leq j \leq n$ sont de dimension $p2^j$. Par conséquent les W^j sont aussi de dimension $p2^j$. D'un point de vue algorithmique, représenter une courbe périodique revient à considérer les indices i modulo $p2^j$ dans $\mathbf{c}_i^j$ et $\mathbf{d}_i^j$.

Parce que très répandues comme modèles d'objets lisses, et pour leurs avantages présentés en section 1.2, nous allons utiliser des courbes B-spline, uniformes et périodiques de période 1. En particulier nos exemples sont basés sur des B-splines quadratiques. Soit $\mathbf{c}(t)$ une telle courbe. D'après (1.6) elle s'écrit :

$$\mathbf{c}(t) = \sum_{i=0}^{p2^n-1} \mathbf{c}_i^n \varphi_i^n(t), \tag{2.1}$$

avec

$$\varphi_i^n(t) = \sum_{k \in \mathbb{Z}} N_i^2(t-k)$$

la fonction B-spline uniforme de degré 2 périodisée (voir annexe A.2). Ce choix est en outre numériquement intéressant, car la base d'ondelettes associée est stable. Par conséquent le banc de filtre est numériquement stable.

La décomposition en ondelettes B-splines quadratiques vérifie le schéma de décomposition suivant, pour $0 < j \leq n$:

$$\mathbf{c}_i^{j-1} = \frac{1}{4}\big(-\mathbf{c}_{2i-2}^j + 3\mathbf{c}_{2i-1}^j + 3\mathbf{c}_{2i}^j - \mathbf{c}_{2i+1}^j\big) \tag{2.2}$$

$$\mathbf{d}_i^{j-1} = \frac{1}{4}\big(\mathbf{c}_{2i-2}^j - 3\mathbf{c}_{2i-1}^j + 3\mathbf{c}_{2i}^j - \mathbf{c}_{2i+1}^j\big), \tag{2.3}$$

et la reconstruction :

$$\mathbf{c}_{2i}^j = \frac{3}{4}\big(\mathbf{c}_i^{j-1} + \mathbf{d}_i^{j-1}\big) + \frac{1}{4}\big(\mathbf{c}_{i+1}^{j-1} - \mathbf{d}_{i+1}^{j-1}\big) \tag{2.4}$$

$$\mathbf{c}_{2i+1}^j = \frac{1}{4}\big(\mathbf{c}_i^{j-1} + \mathbf{d}_i^{j-1}\big) + \frac{3}{4}\big(\mathbf{c}_{i+1}^{j-1} + \mathbf{d}_{i+1}^{j-1}\big). \tag{2.5}$$

Ces équations soulèvent plusieurs remarques qui permettent de comprendre l'origine de ce schéma, ainsi que son intérêt :

i. Dans (2.4) et (2.5) si les détails sont nuls on reconnaît le schéma de subdivision de Chaïkin [Cha74]

$$\mathbf{c}_{2i}^j = \frac{3}{4}\mathbf{c}_i^{j-1} + \frac{1}{4}\mathbf{c}_{i+1}^{j-1} \tag{2.6}$$

$$\mathbf{c}_{2i+1}^j = \frac{1}{4}\mathbf{c}_i^{j-1} + \frac{3}{4}\mathbf{c}_{i+1}^{j-1}, \tag{2.7}$$

connu pour converger vers les B-splines quadratiques uniformes.

ii. Ces équations ont leur correspondance sur les fonctions de base (voir équations (1.3) et (1.4)), sous forme de relation à deux échelles pour les B-splines quadratiques (illustrées Fig. 2.1) :

$$\varphi_i^{j-1} = \frac{1}{4}\varphi_{2i}^j + \frac{3}{4}\varphi_{2i+1}^j + \frac{3}{4}\varphi_{2i+2}^j + \frac{1}{4}\varphi_{2i+3}^j \tag{2.8}$$

$$\psi_i^{j-1} = \frac{-1}{4}\varphi_{2i}^j + \frac{-3}{4}\varphi_{2i+1}^j + \frac{3}{4}\varphi_{2i+2}^j + \frac{1}{4}\varphi_{2i+3}^j. \tag{2.9}$$

Cela signifie que quel que soit le niveau j, les fonctions d'échelle φ^j sont des B-splines quadratiques périodiques, la différence se faisant sur la séquence de nœuds uniforme $\{\frac{i}{p2^j}\}_i$ qui dépend de j.

iii. Ces équations sont invariantes par translation, *i.e.* quelle que soit la position i sur le polygone de contrôle le filtre appliqué est identique. Le schéma est dit *uniforme*. Ceci fait directement écho à l'invariance par translation des fonctions B-splines uniformes. Les matrices A^j, B^j, P^j et Q^j sont alors circulantes (voir annexe A.2), donc il n'est nul besoin de les stocker entièrement.

iv. Ces équations ne dépendent pas de j, ce qui signifie que les matrices A^j, B^j, P^j et Q^j ont la même forme quand j varie, seule leur dimension change. Le schéma est dit "stationnaire".

v. Les filtres sont locaux (seulement 4 coefficients interviennent dans chaque transformation), donc les matrices A, B, P et Q sont creuses. Cela correspond au support local des fonctions B-splines mentionné en section 1.2.

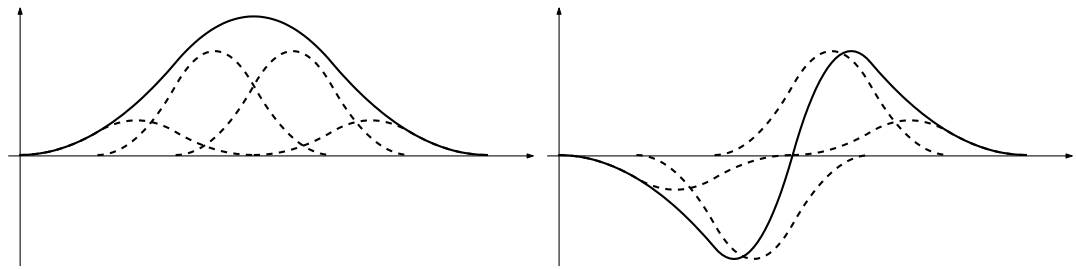


FIG. 2.1 – Ondelettes B-spline quadratiques uniformes
Décomposition de la fonction d'échelle φ_0^{j-1} (à gauche) et de l'ondelette ψ_0^{j-1} (à droite) avec φ_i^j, $i = 0, 1, 2, 3$. Ceci correspond aux expressions (2.8) (à gauche) et (2.9) (à droite) : φ_0^{j-1} et ψ_0^{j-1} sont en trait plein ; les φ_i^j pondérés sont en traits pointillés.

2.3 Calcul d'aire

Soit $\mathbf{c}(t) = \big(x(t), y(t)\big)$ une courbe paramétrique plane et périodique définie par (2.1). Nous souhaitons la déformer en conservant l'aire qu'elle délimite. Pour cela il est nécessaire de pouvoir évaluer l'aire quel que soit le niveau de résolution de la courbe. La section 2.3.1 présente les formules multirésolution permettant de calculer l'aire. Dans la section 2.3.2 on explique comment le calcul peut être effectué de manière efficace en développant une relation de récurrence.

2.3.1 Aire délimitée par une courbe multirésolution

En supposant que $\mathbf{c}$ est de classe C^1 (ce qui est le cas pour les splines quadratiques uniformes), l'aire signée incluse dans $\mathbf{c}(t)$ est donnée par le théorème de Green [Elb00, EL91] :

$$\mathcal{A} = \frac{1}{2} \oint x(t)y'(t) - x'(t)y(t)dt. \tag{2.10}$$

Dans notre cas la courbe s'exprime dans la base décomposée par (voir (1.7)) :

$$\mathbf{c}(t) = (\mathbf{c}^e)^T(\varphi^e(t)) + (\mathbf{d}^e)^T(\psi^e(t)) + \ldots + (\mathbf{d}^{n-1})^T(\psi^{n-1}(t)). \tag{2.11}$$

Avant de continuer fixons quelques notations. Introduisons la forme bilinéaire anti-symétrique

$$I(\Gamma, \Theta) = \begin{pmatrix} & I(\gamma_{i_1}, \theta_{i_2})_{i_1 i_2} & \end{pmatrix}, \tag{2.12}$$

où $\Gamma = \big(\gamma_{i_1}\big)_{i_1}$ et $\Theta = \big(\theta_{i_2}\big)_{i_2}$ sont deux vecteurs colonne de fonctions périodiques et

$$I(\gamma_{i_1}, \theta_{i_2}) = \oint \gamma_{i_1}(t)\theta'_{i_2}(t) - \gamma'_{i_1}(t)\theta_{i_2}(t)dt. \tag{2.13}$$

Pour un niveau de décomposition e donné, tous les coefficients multirésolution ($\mathbf{c}_i^e$, $0 \leq i < p2^e$, et $\mathbf{d}_i^j$, $e \leq j < n$, $0 \leq i < p2^j$) sont regroupés dans un vecteur de $(\mathbb{R}^2)^N$. Nous notons $X^e \in \mathbb{R}^N$ et $Y^e \in \mathbb{R}^N$ les vecteurs première et seconde coordonnée de ce vecteur :

$$\begin{pmatrix} & X^e,\ Y^e & \end{pmatrix} = \begin{pmatrix} \mathbf{c}^e \\ \mathbf{d}^e \\ \mathbf{d}^{e+1} \\ \vdots \\ \mathbf{d}^{n-1} \end{pmatrix}.$$

Réinjectons maintenant l'expression (2.11) dans l'équation (2.10), on obtient l'évaluation de l'aire à n'importe quel niveau de résolution $e \leq n$:

$$2\mathcal{A} = (X^e)^T \left[M^e \right] (Y^e), \quad \forall e \in \{0, \dots, n\}, \tag{2.14}$$

où

$$M^e = \begin{bmatrix} I(\varphi^e, \varphi^e) & I(\varphi^e, \psi^j)_{j=e}^{n-1} \\ I(\psi^k, \varphi^e)_{k=e}^{n-1} & I(\psi^k, \psi^j)_{k,j=e}^{n-1} \end{bmatrix} = \begin{bmatrix} A & B \\ -B^T & C \end{bmatrix} \tag{2.15}$$

est une matrice anti-symétrique de taille $p2^n \times p2^n$ contenant 4 blocs principaux (voir Fig. 2.2) :

- Le bloc A (jaune) est de taille $p2^e \times p2^e$ et contient $I(\varphi^e, \varphi^e)$.
- Le bloc B (rouge) est de taille $p2^e \times (p2^n - p2^e)$ et contient $I(\varphi^e, \psi^j)$ pour $j = e, \dots, n-1$.
- Le bloc C (vert) est de taille $(p2^n - p2^e) \times (p2^n - p2^e)$ et contient $I(\psi^k, \psi^j)$ pour $e \leq j, k \leq n-1$.

La taille des blocs varie en fonction du niveau de résolution e (Fig. 2.3) mais la taille totale de M^e est toujours la même.

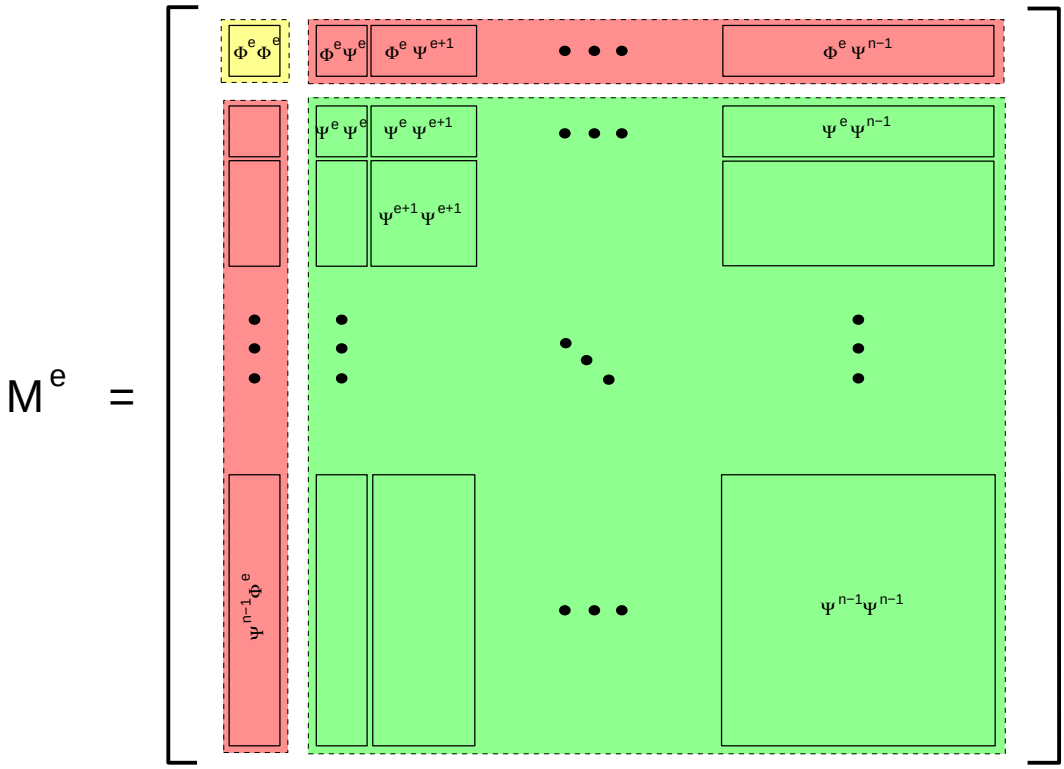


Fig. 2.2 – Matrice d'aire.
Le bloc A (Eq. (2.15)), est en jaune en haut à gauche. Les blocs B et $-B^T$ sont en rouge. Le bloc C est en vert en bas à droite.

2.3.2 Calcul efficace des matrices d'aire

Les matrices d'aire $(M^j)_{0 \leq j \leq n}$ peuvent être pré-calculées une fois pour toutes les déformations effectuées sur la courbe. Pour chacun des $n+1$ niveaux de résolution, la matrice contient

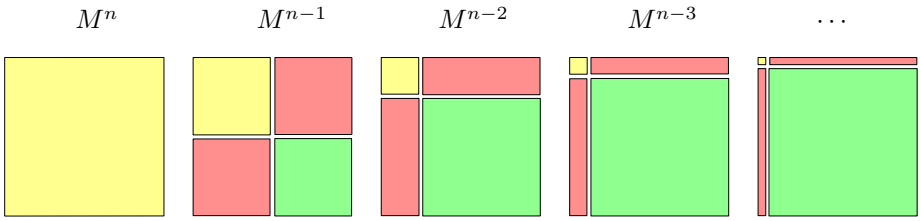


FIG. 2.3 – Matrices d'aire M^e pour $e \in \{0, \ldots, n\}$.
Elles ont la même taille $p2^n \times p2^n$ mais leur décomposition en quatre blocs est différente selon le niveau de résolution e.

N^2 éléments (avec $N = p2^n$) qui sont des intégrales du type (2.13). Le calcul direct de toutes les matrices dans leur forme (2.15) nécessiterait un total de $O\big(nN^2\big) = O\big(N^2 \log(N)\big)$ intégrations. Dans le cas des B-splines, comme dans les autres cas où l'on connaît une expression analytique des fonctions de base, il est possible de faire de l'intégration formelle, sinon il faut envisager l'utilisation de formules de quadrature. Ces évaluations s'avèrent inutilement coûteuses : nous allons montrer ici qu'il est possible de ne calculer que la matrice M^n à l'aide des intégrales. Nous développons ensuite une formule de récurrence sur les matrices de niveaux inférieurs à l'aide des filtres de synthèse.

Formules de récurrence.

D'après les équations (1.3), les fonctions d'échelle et les ondelettes au niveau $j-1$ peuvent s'exprimer comme combinaison linéaire des fonctions d'échelle au niveau j. Par bilinéarité de la fonctionnelle (2.13), les éléments de la matrice M^{e-1} sont calculables récursivement à partir de ceux de M^e.

Définissons pour cela les filtres matriciels (**P**), (**Q**), (**PP**), (**PQ**) et (**QQ**) à partir des filtres de synthèse P^j et Q^j. Ces filtres permettront de calculer M^{e-1} par sous-blocs à partir des sous-blocs de M^e. Nous entendons par sous-bloc un morceau (de type (2.12)) de la matrice M^e. Observons la forme de M^e figure 2.2. Pour passer à M^{e-1}, le bloc C n'est pas modifié. Chaque sous-bloc $I(\varphi^e, \psi^k)$ du bloc B est décomposé en deux sous-blocs $I(\varphi^{e-1}, \psi^k)$ et $I(\psi^{e-1}, \psi^k)$. Le deuxième argument (*i.e.* ψ^k) de la forme bilinéaire I est invariant, et l'on peut définir :

$$\textbf{filtre (P) :} \quad I(\varphi^{e-1}, \psi^k) = (P^e)^T\, I(\varphi^e, \psi^k) \tag{2.16}$$
$$\textbf{filtre (Q) :} \quad I(\psi^{e-1}, \psi^k) = (Q^e)^T\, I(\varphi^e, \psi^k) \tag{2.17}$$

Par symétrie on obtient des formules analogues dans le cas où le premier argument est invariant (bloc $-B^T$), mais comme M est antisymétrique nous ne l'utilisons pas. Le bloc A contient quant à lui un seul sous-bloc $I(\varphi^e, \varphi^e)$ qui est décomposé en quatre sous-blocs $I(\varphi^{e-1}, \varphi^{e-1})$, $I(\varphi^{e-1}, \psi^{e-1}) = -I(\psi^{e-1}, \varphi^{e-1})^T$, et $I(\psi^{e-1}, \psi^{e-1})$, ce qui nous amène à définir :

$$\textbf{filtre (PP) :} \quad I(\varphi^{e-1}, \varphi^{e-1}) = (P^e)^T\, I(\varphi^e, \varphi^e)\, (P^e) \tag{2.18}$$
$$\textbf{filtre (PQ) :} \quad I(\varphi^{e-1}, \psi^{e-1}) = (P^e)^T\, I(\varphi^e, \varphi^e)\, (Q^e) \tag{2.19}$$
$$\textbf{filtre (QQ) :} \quad I(\psi^{e-1}, \psi^{e-1}) = (Q^e)^T\, I(\varphi^e, \varphi^e)\, (Q^e) \tag{2.20}$$

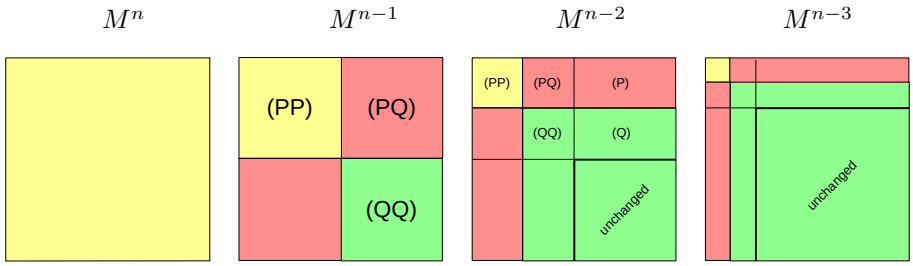


Fig. 2.4 – Calcul récursif de la matrice d'aire.
M^{n-1} à partir de M^n, M^{n-2} à partir de M^{n-1} etc.

Algorithme récursif.

À partir de ces filtres nous pouvons définir l'algorithme récursif de calcul des matrices schématisé figure (2.4). M^{e-1} est calculée à partir de M^e en appliquant les filtres **(P)** et **(Q)** sur le bloc B, et les filtres **(PP)**, **(PQ)** et **(QQ)** sur le bloc A. Le bloc C est invariant, ce qui allège significativement les calculs. Il y a plusieurs avantages à utiliser cet algorithme :

- Il n'est plus nécessaire de pré-calculer et de stocker toutes les matrices. M^e est calculée rapidement quand le niveau de décomposition de la courbe change au cours du processus d'édition.
- Le calcul peut être effectué en place, il n'est pas besoin d'allocation mémoire supplémentaire.
- La stabilité du calcul est assurée par la stabilité du banc de filtre, c'est-à-dire par la stabilité de la base d'ondelettes.

Complexité.

Le calcul de la complexité asymptotique de cet algorithme dépend de la base d'ondelettes choisie. Si les matrices P^e, Q^e et M^e sont au format plein, le nombre de multiplications de flottants pour passer de M^e à M^{e-1} est :

- Application de **(P)** et **(Q)** : deux produits matriciels $(p2^{e-1})\times(p2^e)$ par $(p2^e)\times(p2^n-p2^e)$, c'est-à-dire $p^4(2^e)^3(2^n-2^e)$ opérations.
- Application de **(PP)**, **(PQ)** et **(QQ)** : quatre produits matriciels $(p2^{e-1})\times(p2^e)$ par $(p2^e)\times(p2^e)$, c'est-à-dire $2p^4(2^e)^4$ opérations.

Ce qui fait un total de $p^4(2^e)^3(2^n+2^e)$ multiplications. Par conséquent le calcul de M^0 à partir de M^n coûte

$$p^4 2^n \sum_{e=1}^{n} 8^e + p^4 \sum_{e=1}^{n} 16^e = p^4 2^n \frac{8}{7}(8^n-1) + p^4 \frac{16}{15}(16^n-1),$$

c'est-à-dire $O(N^4)$. Ceci semble élevé par rapport à $O(N^2\log(N))$ évaluations d'intégrale. Pour améliorer cette borne, nous allons profiter du support compact des fonctions de base.

Dans le cas d'ondelettes à support compact (comme les ondelettes B-spline), les matrices P^e et Q^e sont creuses, avec $O(p2^e)$ coefficients non nuls. Si de plus la base est invariante par translation (voir section 2.2), ces matrices sont circulantes, et ne sont donc pas stockées sous forme matricielle, mais sous la forme d'une suite de quelques coefficients. Idéalement, il faudrait de plus

prendre en compte le fait que les matrices M^e sont elles-mêmes creuses. M^n a $O(N)$ coefficients non nuls seulement, mais le remplissage des autres M^e est difficile à borner précisément.

Afin de faire une majoration grossière, considérons le cas où elles sont au format plein, et seuls les filtres sont codés au format creux. Le coût de l'application des filtres est alors de l'ordre de la taille du bloc :

- Application de **(P)** et **(Q)** : $O\big(p^2 2^e (2^n - 2^e)\big)$ opérations.
- Application de **(PP)**, **(PQ)** et **(QQ)** : $O\big((p2^e)^2\big)$ opérations.

Donc le coût total du calcul de M^0 à partir de M^n est en $O\big(N^2 \log(N)\big)$, ce qui assure un résultat rapide pour quelques milliers de points.

Rappelons que le calcul direct des coefficients nécessite $O\big(N^2 \log(N)\big)$ évaluations d'intégrales. Pour comprendre l'avantage du calcul récursif, il faut prendre en compte le coût du calcul d'une intégrale : les B-splines sont des polynômes de degré d par morceaux sur $d+1$ intervalles. Une seule intégration formelle coûte donc bien plus cher que les additions et les multiplications nécessaires pour appliquer un filtre. En outre, la méthode récursive est simple à implémenter car les filtres sont déjà construits pour manipuler la courbe.

2.4 Déformation multirésolution

La contrainte d'aire pour une courbe fermée multirésolution (2.11) est maintenant exprimée à n'importe quel niveau de résolution par la forme bilinéaire (2.14), et les calculs peuvent être faits efficacement par la méthode présentée section 2.3.2. Nous présentons ici une méthode de déformation à une échelle quelconque qui intègre la contrainte d'aire constante.

Grâce à l'expression multirésolution de la contrainte d'aire, nous allons pouvoir construire une boucle d'édition de type FIG. 2.5. Il sera alors possible interactivement de :

- modifier la courbe localement ou globalement à n'importe quel niveau de résolution,
- préserver les détails ou pas,

tout en conservant l'aire délimitée par la courbe.

2.4.1 Principe

La méthode de déformation est composée de trois étapes principales (FIG. 2.5) :

i. **décomposition :** La courbe est décomposée jusqu'à une échelle e qui a été définie par l'utilisateur. La courbe est alors représentée par un polygone de contrôle et un ensemble de coefficients de détail (voir formule(2.11)).

ii. **déformation :** L'utilisateur définit une déformation en déplaçant un point du polygone de contrôle.

iii. **conservation d'aire :** À chaque mouvement de souris, une nouvelle courbe est calculée et affichée suivant la déformation définie par l'utilisateur et en préservant l'aire intérieure de la courbe initiale. Cette étape constitue le problème que nous nous proposons de résoudre.

Comme nous l'avons souligné dans l'introduction, il peut être utile, en fin de boucle, de reconstruire la courbe au niveau fin, mais cela n'est pas indispensable. Une fois la courbe décomposée (i), il suffit d'itérer les étapes (ii) et (iii) pour établir une suite de déformations à aire constante.

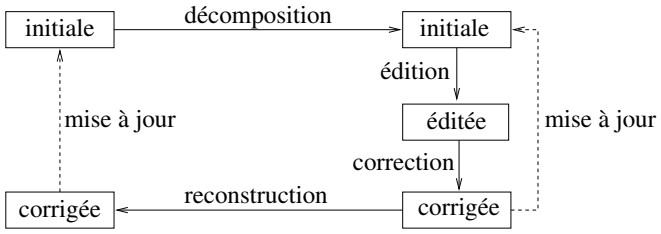


Fig. 2.5 – Boucle d'édition pour la conservation d'aire.

Étant donnée la courbe multirésolution $\mathbf{c}(t)$ et fixé le niveau de décomposition e auquel est appliquée la déformation, nous négligerons l'indice e implicite, et utiliserons les notations suivantes :

- $\mathcal{A}_{ref}$ est l'aire de référence que nous voulons conserver,
- (X_0, Y_0) sont les coordonnées après déformation du polygone de contrôle, et avant les corrections pour conserver l'aire,
- $\mathcal{A}_0 = \frac{1}{2} X_0^T M Y_0$ est l'aire correspondante qui doit être corrigée,
- (X, Y) sont les coordonnées après correction,
- $\mathcal{A} = \frac{1}{2} X^T M Y$ est la nouvelle aire, telle que $\mathcal{A} = \mathcal{A}_{ref}$.

Nous pouvons alors reformuler les trois étapes en ces termes :

i. exprimer $\mathbf{c}(t)$ dans une base multirésolution de V^n,

ii. déformer le polygone de contrôle pour obtenir X_0 et Y_0,

iii. calculer X et Y "proches de" X_0 et Y_0 tels que $\mathcal{A} = \mathcal{A}_{ref}$.

Nous remarquons qu'il y a seulement une contrainte scalaire ($\mathcal{A} = \mathcal{A}_{ref}$) pour $p2^n$ degrés de liberté vectoriels $(\mathbf{c}^e, \mathbf{d}^e, \ldots, \mathbf{d}^{n-1})$, ce qui ouvre de nombreuses possibilités pour définir la correction d'aire. Afin de proposer une méthode aussi générale que possible, nous avons choisi une méthode d'optimisation. Plus précisément, il s'agit de la minimisation d'une fonction objectif sous contrainte d'aire. La fonction objectif contient un terme de distance qui assure un résultat proche de la courbe éditée, et un terme de lissage qui permet d'obtenir un résultat visuellement lisse.

2.4.2 Fonction objectif

Le premier critère d'optimisation est la proximité entre (X, Y) et (X_0, Y_0). Nous choisissons la distance quadratique, car elle est facile à manipuler :

$$\mathcal{D}(X, Y) = \|X - X_0\|^2 + \|Y - Y_0\|^2 \ .$$

Le deuxième critère d'optimisation est la régularité. En design variationnel, la notion de courbe ou de surface "lisse", au sens visuel et non analytique, est décrite par un modèle physique [NR83, WW92]. Le critère plus utilisé provient de l'observation de membranes élastiques fines : elles minimisent l'énergie de tension et ont une forme visuellement agréable. L'énergie de tension d'une courbe paramétrique est définie à partir de la courbure κ :

$$\mathcal{E}_B = \int \kappa^2(t) dt \ .$$

Cette expression n'est pas évidente à manipuler parce qu'elle est non linéaire. Suivant une méthode classique [FRSW87, BH94] nous allons utiliser une version linéarisée :

$$\mathcal{E} = \int |\mathbf{c}"(t)|^2 dt = \int x"(t)^2 + y"(t)^2 dt \ .$$

Cette approximation est égale à l'énergie exacte dans le cas d'une paramétrisation normale $|\mathbf{c}'(t)| \equiv 1$.

Dans le cadre multirésolution qui nous concerne, $\mathcal{E}$ doit être évaluée pour une courbe exprimée par (2.11). De façon tout à fait similaire au calcul d'aire (2.14), l'énergie peut s'exprimer par la forme quadratique suivante :

$$\mathcal{E}(X,Y) = \frac{1}{2}\big((X^e)^T H^e X^e + (Y^e)^T H^e Y^e\big), \tag{2.21}$$

où H^e est une matrice symétrique définie positive qui est calculée de la même manière que M^e (se référer à la section 2.3.2) à partir de la matrice creuse circulante H^n (détaillée en annexe A.2 pour les B-splines quadratiques).

La fonction objectif peut maintenant s'exprimer

$$(1-\beta)\mathcal{E}(X,Y) + \beta\mathcal{D}(X,Y) \ ,$$

où $\beta \in [0,1]$ est un paramètre de contrôle qui permet de choisir un résultat plus lisse (pour β petit, $\mathcal{E}$ est prépondérant) ou plus proche de la courbe éditée (pour β proche de 1, $\mathcal{D}$ est prépondérant). L'impact de β est discuté sur un exemple dans la prochaine section.

2.4.3 Minimisation sous contrainte

Le problème de conservation d'aire, consistant à définir la courbe finale (X,Y) à partir de la courbe éditée (X_0,Y_0), peut maintenant se formuler sous la forme d'une minimisation sous contrainte :

$$\min_{X,Y} \quad (1-\beta)\mathcal{E}(X,Y) + \beta\mathcal{D}(X,Y) \quad \text{sous contrainte} \quad \mathcal{A} = \mathcal{A}_{ref},$$

qui peut être transformée en problème minimax en utilisant la méthode des multiplicateurs de Lagrange [GW73, Cia88] :

$$\max_{\lambda} \min_{X,Y} \quad (1-\beta)\mathcal{E}(X,Y) + \beta\mathcal{D}(X,Y) + \lambda(X^T M Y - 2\mathcal{A}_{ref}) \tag{2.22}$$

où λ est le multiplicateur de Lagrange pour la contrainte $X^T M Y - 2\mathcal{A}_{ref}$.

La contrainte étant quadratique, le lagrangien (2.22) est cubique (par rapport à X, Y et λ). Or la résolution est plus efficace si le lagrangien est quadratique, car on peut exprimer le problème sous forme d'un système linéaire. C'est pourquoi nous proposons de linéariser la contrainte d'aire. Une première solution, exposée dans [Elb00], consiste à briser la symétrie des coordonnées en fixant alternativement X et Y à chaque petit déplacement, et à résoudre (2.22) respectivement en (Y,λ) et (X,λ). Nous reviendrons en section 2.5.2 sur cette solution. Afin de conserver la symétrie nous proposons plutôt d'approximer la contrainte $\mathcal{A} = \mathcal{A}_{ref}$ par :

$$X_0^T M Y + X^T M Y_0 = 2(\mathcal{A}_0 + \mathcal{A}_{ref}) \ .$$

Il s'agit bien d'une approximation dans la mesure où, si l'on pose $X = X_0 + \delta X$ et $Y = Y_0 + \delta Y$, on obtient

$$2\mathcal{A} = 2\mathcal{A}_{ref} + \delta X^T M \delta Y \ .$$

Par conséquent, si $\delta X^T M \delta Y$ est proche de 0, *i.e.* petit devant $X^T M \delta Y$ et $\delta X^T M Y$, alors l'aire $\mathcal{A} \approx \mathcal{A}_{ref}$ sera bien approximée. Cette linéarisation est en fait le développement limité au premier ordre en δx_i et δy_i (qui composent δX et δY, pour $0 \leq i < p2^n$) au voisinage de 0.
En utilisant la contrainte approximée à la place de l'exacte dans (2.22), le problème devient

$$\max_{\lambda} \min_{X,Y} \ g(X, Y, \lambda) \tag{2.23}$$

avec le lagrangien

$$g(X, Y, \lambda) = (1 - \beta)\mathcal{E}(X, Y) + \beta\mathcal{D}(X, Y) + \lambda\big(X_0^T M Y + X^T M Y_0 - 2(\mathcal{A}_0 + \mathcal{A}_{ref})\big). \tag{2.24}$$

Une condition nécessaire pour que le triplet (X, Y, λ) soit solution de (2.23) est d'annuler le gradient $\vec{\nabla} g$, ce qui revient à résoudre le système linéaire suivant :

$$\begin{bmatrix} H + 2\beta\mathrm{Id} & 0 & MY_0 \\ 0 & H + 2\beta\mathrm{Id} & M^T X_0 \\ Y_0^T M^T & X_0^T M & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ \lambda \end{bmatrix} = \begin{bmatrix} 2\beta X_0 \\ 2\beta Y_0 \\ 2(\mathcal{A}_{ref} + \mathcal{A}_0) \end{bmatrix} \tag{2.25}$$

Si ce système est de rang plein, alors la solution est unique, et c'est la solution du problème de minimisation.

Résolution du système linéaire.

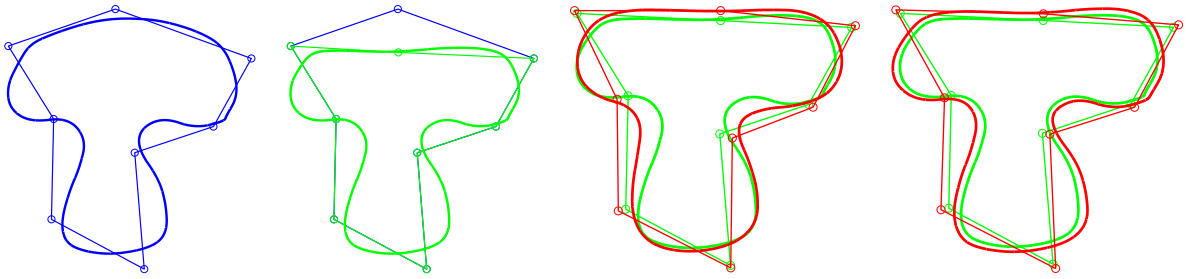
Le système linéaire (2.25) étant creux, il est judicieux d'utiliser une méthode de résolution itérative. De plus il est symétrique, mais à cause des blocs MY_0 et $M^T X_0$ on ne peut pas assurer qu'il est défini positif. Par conséquent nous avons choisi la méthode du gradient bi-conjugué [PVTF02], qui fonctionne même pour les systèmes non symétriques.

Itération de la résolution.

La solution obtenue par la résolution du système (2.25) est la solution exacte de (2.23), donc elle vérifie la contrainte d'aire linéarisée, mais elle approxime la contrainte d'aire exacte. Pour augmenter la précision jusqu'à un seuil quelconque il est possible d'itérer la résolution :

tant que $|\mathcal{A}_0 - \mathcal{A}_{ref}| > seuil * |\mathcal{A}_{ref}|$ **faire**
  calculer X et Y solution de (2.25)
  $(X_0, Y_0) \leftarrow (X, Y)$
fin

Nous n'avons pas de preuve formelle de la convergence de ce processus. Les résultats empiriques sont néanmoins convaincants : pour un seuil de 10^{-5} et quelques milliers de points, il faut moins d'une dizaine d'itérations.



(a) courbe initiale (b) courbe éditée (c) courbe finale $\beta = 0.5$ (d) $\beta = 0.99$

FIG. 2.6 – Influence du coefficient β.
La courbe initiale ($n = 5$) et son polygone de contrôle ($e = 3$) sont déformés par déplacement du point de contrôle supérieur (en vert/clair).
L'aire est corrigée (en rouge/sombre) avec $\beta = 0.5$ et $\beta = 0.99$.

Influence du coefficient β.

La figure 2.6 illustre les différentes étapes de l'algorithme. La courbe initiale (bleue) est définie par 32 points de contrôle ($n = 5$). Après deux étapes de décomposition on obtient 8 coefficients d'échelle $\mathbf{c}^3$ (polygone de contrôle bleu), 8 coefficients de détails $\mathbf{d}^3$ au niveau 3, et 16 coefficients de détails $\mathbf{d}^4$ au niveau 4, qui représentent la courbe dans la base $(\varphi^3, \psi^3, \psi^4)$ de $V^5 = V^3 \oplus W^3 \oplus W^4$. La déformation est définie par le déplacement d'un point de contrôle (polygone vert). Grâce à la représentation multirésolution, la modification de ce coefficient grossier influence toute la partie supérieure de la courbe. Si l'on reconstruit la courbe fine (courbe verte) on s'aperçoit que l'aire a diminué. Deux courbes (en rouge) vérifiant (2.23) sont calculées sous forme décomposée, puis reconstruites. L'une est paramétrée par $\beta = 0.5$, l'autre par $\beta = 0.99$. On observe qu'une plus petite valeur de β donne un résultat plus lisse, alors qu'une plus grande valeur donne un résultat plus proche de la courbe avant correction (verte). Le choix de la valeur de β est laissée à la discrétion de l'utilisateur, pour contrôler le type de résultats qu'il souhaite obtenir.

2.4.4 Minimisation partielle

La méthode présentée dans la section 2.4.3 appelle un certain nombre de remarques :

i. Il s'agit d'une optimisation globale, donc nécessairement *toute* la courbe sera modifiée pour la correction d'aire. Cependant il semble intéressant de pouvoir modifier la courbe localement.

ii. En particulier le point de contrôle déplacé par l'utilisateur peut être modifié lors de la correction : sur la figure 2.6 le point de contrôle supérieur n'est pas conservé exactement entre la courbe verte et la courbe rouge. Autrement dit la déformation, qu'elle résulte d'une édition directe par l'utilisateur ou d'une contrainte externe, n'est pas respectée scrupuleusement.

iii. La minimisation de l'énergie de tension (2.21), qui permet de contrôler une certaine régularité de la courbe, a l'inconvénient d'altérer petit à petit les détails fins, porteurs de beaucoup d'énergie (voir FIG. 2.8). Or un grand intérêt de la représentation multirésolution, dont nous avons parlé en introduction, est de pouvoir modifier la forme globale

sans modifier ces détails qui caractérisent la courbe.

Pour palier à ces trois désagréments, nous proposons de faire une minimisation partielle. Le lagrangien (2.24) est conservé, mais les coefficients de la décomposition qui se trouvent dans X et Y sont séparés en deux parties : une partie fixe et une partie variable. La partie fixe regroupe les coefficients que l'on ne souhaite pas voir corrigés et la partie variable regroupe ceux qui serviront à corriger l'aire. C'est donc par rapport à ces derniers seuls que la dérivée partielle du lagrangien (2.24) est annulée.

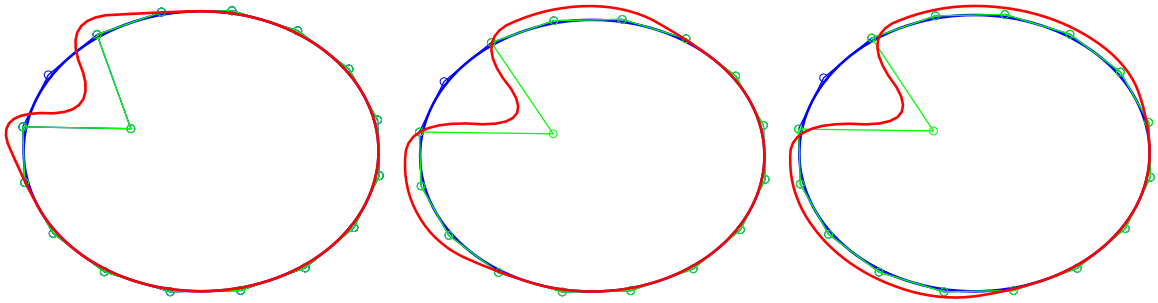


Fig. 2.7 – Correction locale.
La courbe initiale (bleu) est construite avec $n = 7$ et décomposée au niveau $e = 4$. Un point de contrôle est édité (polygone vert) puis l'aire est corrigée sur un voisinage de 1, 3 et 5 points (de gauche à droite). Seule la courbe fine résultat est représentée (en rouge).

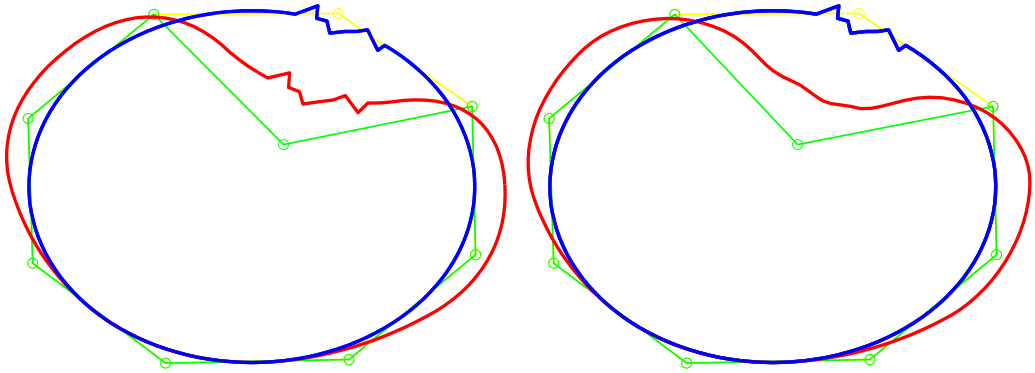


Fig. 2.8 – Conservation des détails.
La courbe initiale (bleue) subit deux fois la même déformation (en vert) au niveau $e = 3$. La courbe finale (rouge) est obtenue avec (à gauche) et sans (à droite) conservation des détails.

La répartition des coefficients selon ces deux catégories (fixe et variable) permet de résoudre les problèmes cités ci-dessus :

i. En laissant variables les coefficients grossiers voisins du point édité, et en fixant les plus éloignés, on peut contrôler l'étendue de la correction. La figure 2.7 illustre cette sélection : la courbe initiale subit trois fois la même déformation mais l'aire est corrigée sur une partie plus ou moins grande.

ii. En fixant le point de contrôle qui est édité par l'utilisateur, la déformation est strictement

observée. Cette remarque peut sembler mineure alors qu'elle est capitale : sans elle il est impossible d'assurer strictement la position d'un point. Dans un processus d'édition drag-and-drop cela signifierait que le point sélectionné ne reste pas sous le pointeur de la souris !

iii. Selon que l'on met les coefficients de détail des différentes échelles dans la partie fixe ou dans la partie variable, ils seront conservés ou pas. La courbe FIG. 2.8 présente des détails fins localisés. Elle est déformée en fixant les détails (à gauche) ou en les laissant variables (à droite). Dans le premier cas ils sont intacts, alors que dans le deuxième ils sont lissés par la minimisation.

Il nous faut ajouter un avantage important de cette sélection des coefficients : la taille du système à résoudre est largement réduite. Celui-ci est en effet défini par :

$$\frac{\partial g}{\partial x_i} = 0 \quad \text{et} \quad \frac{\partial g}{\partial y_i} = 0 \qquad \text{pour } (x_i, y_i) \text{ variable.}$$

C'est un système du même type que (2.25) mais plus petit, avec un membre de droite complexifié par les coefficients fixés. La résolution du système est beaucoup plus rapide. À titre d'exemple, pour la figure 2.8 de droite, l'aire est corrigée sur uniquement deux coefficients voisins de chaque coté, à multiplier par deux coordonnées, plus la ligne de contrainte : le système est de taille 9×9 au lieu de 65×65.

2.4.5 Résultats

Tous les éléments de notre méthode sont maintenant en place. Nous avons vu la façon de calculer l'aire d'une courbe multirésolution fermée, comment l'intégrer dans un processus d'édition, leurs aspects algorithmiques, et nous avons étudié les différentes potentialités que nous offre la représentation multirésolution. Nous présentons ici des résultats plus complexes qui mettent en avant les possibilités. Ces exemples ont été créés par de simples opérations interactives de drag-and-drop via notre éditeur. Les *frame rates* sont de l'ordre de 150 Hz pour 500 points, de 20 à 70 Hz pour 2000 points, et de 10 à 30 Hz pour 4000 points sur un PC standard. Les variations sont dues au coût de la résolution du système qui varie en fonction du remplissage de la matrice, lui même dépendant du niveau de résolution auquel la courbe est édité.

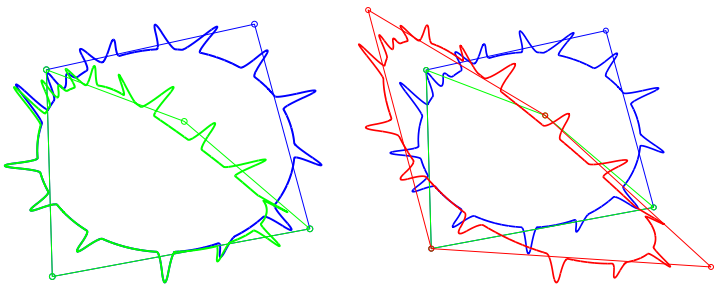


FIG. 2.9 – Déformation avec conservation des détails.
La courbe initiale (bleu) avec 128 points ($n = 7$) est déformée au niveau $e = 2$ (vert) avec conservation d'aire et détails fixés (rouge).

L'intérêt de la formulation en base multirésolution est mis en valeur par la figure 2.9 : l'aire est adaptée en ne corrigeant que les deux voisins directs sur le polygone de contrôle, et sans toucher aux détails. On remarque que la forme générale est modifiée pour l'adaptation d'aire, mais que tous les détails qui caractérisent la courbe sont soigneusement conservés.

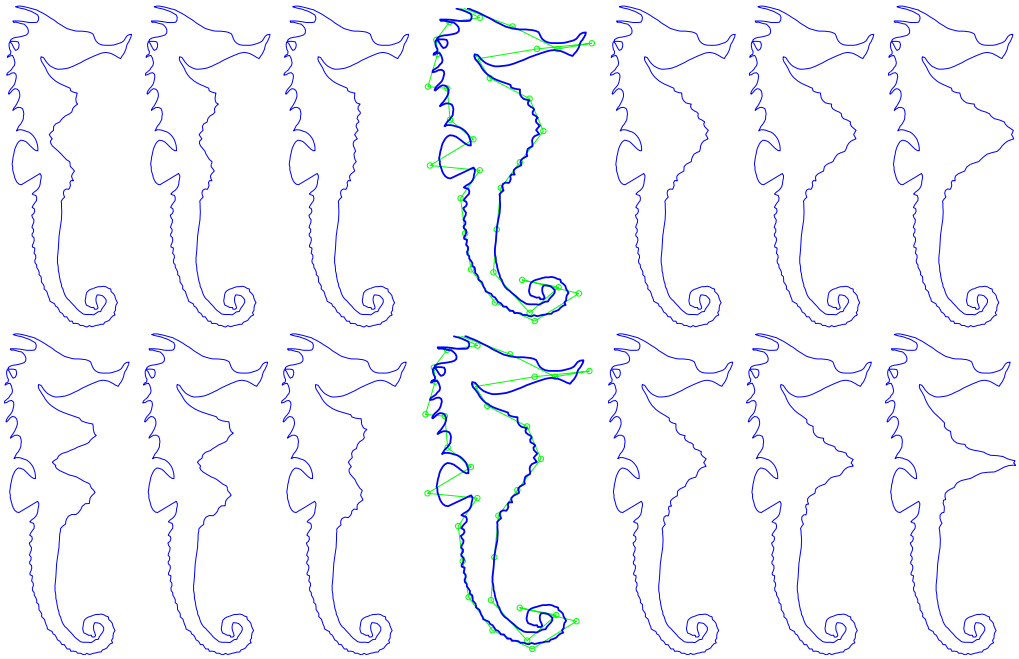


Fig. 2.10 – Déformation multirésolution d'un hippocampe.
La courbe originale (au milieu) a été déformée en déplaçant un point de contrôle au niveau de l'abdomen. En haut : sans conservation d'aire. En bas : avec conservation d'aire.

La figure 2.10 présente une courbe de 512 points avec des détails à des échelles variées. Les deux séries de déformation sont obtenues en déplaçant un point de contrôle de l'abdomen vers la gauche et vers la droite. L'aire est conservée sur la ligne du bas, créant des déformations plus "naturelles" que celle du haut.

Le hérisson de la figure 2.11 est un exemple typique de courbe contenant de nombreux détails très fins, raison pour laquelle la représentation multirésolution est utile. On peut comparer le hérisson allongé puis rétréci avec et sans conservation d'aire. Sur cet exemple, on voit que la contrainte apporte un réalisme, en imitant le comportement réel du hérisson.

2.5 Cas des B-splines non uniformes

Dans des travaux récents [Elb00, Elb01], Elber présente des résultats très proches de ceux que nous venons de présenter. Il y développe une méthode de déformation *multi-échelle* de courbes splines non uniformes à aire constante. Dans la section 2.5.1 nous présentons sa méthode en détail, pour ensuite la comparer à la notre, dans la section 2.5.2. À la lumière de cette comparaison nous apporterons une nuance entre la notion de multi-échelle et la notion de multi-résolutionprécédemment utilisée, et défini en section 1.3.

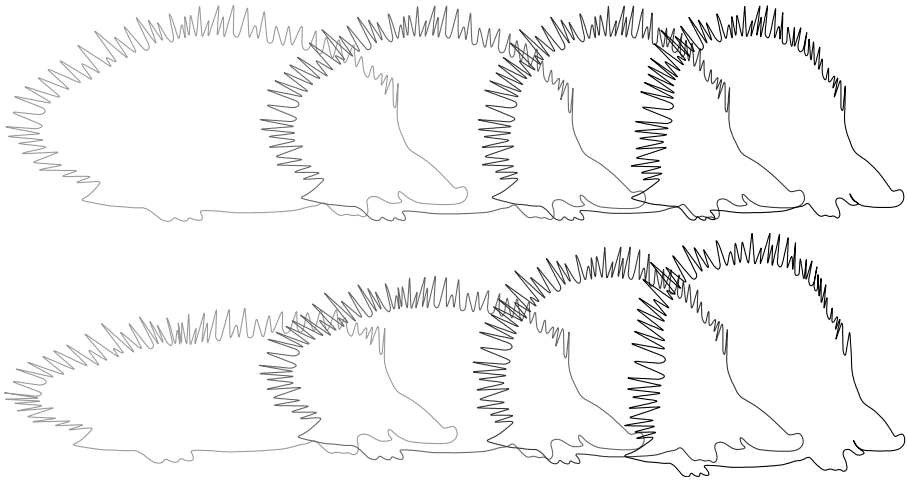


FIG. 2.11 – Déformation multirésolution d'un hérisson.
En haut : déformations sans contrainte. En bas : avec conservation d'aire.

2.5.1 La méthode de l'article [Elb01]

Déformation multi-échelle d'une courbe B-spline.

Soit $\mathbf{c}(t)$ une courbe B-spline de degré d définie comme (1.1) :

$$\mathbf{c}(t) = \sum_{i=0}^{m-1} \mathbf{c}_i N_i^d(t) \ ,$$

avec $\mathbf{c}_i = (x_i, y_i) \in \mathbb{R}^2$, sur une séquence de nœuds $\mathbf{t} = \{t_0, t_1, \ldots, t_{m+d}\}$. L'espace V engendré par les fonctions de base N_i^d est de dimension m dans $\mathbb{R}^2$.

Afin de définir une déformation sur $\mathbf{c}(t)$ à une échelle grossière, on définit un espace $\widetilde{V} \subset V$ de dimension $\widetilde{m}$, engendré par des B-splines $\widetilde{N}_i^d$. Ces fonctions sont définies sur une séquence de nœuds $\widetilde{\mathbf{t}} = \{\widetilde{t}_0, \ldots, \widetilde{t}_{\widetilde{m}+d}\} \subset \mathbf{t}$ telle que :

- les nœuds de bord sont conservés, *i.e.* $\widetilde{t}_i = t_i$ et $\widetilde{t}_{\widetilde{m}+i} = t_{m+i}$ pour $i \in \{0, \ldots, d\}$,
- une partie des nœuds intérieurs est supprimée, *i.e.* $\{\widetilde{t}_{d+1}, \ldots, \widetilde{t}_{\widetilde{m}-1}\} \subsetneq \{t_{d+1}, \ldots, t_{m-1}\}$.

Soit $\Delta(t)$ un élément de $\widetilde{V}$, appelé déformation. L'opération $\bar{\mathbf{c}} = \mathbf{c} + \Delta$ déforme la courbe à une échelle d'autant plus grossière que $\widetilde{\mathbf{t}}$ contient peu de nœuds. Grâce à l'algorithme d'insertion de nœuds [Far01], la déformation

$$\Delta(t) = \sum_{j=0}^{\widetilde{m}} \widetilde{\delta}_j \widetilde{N}_j(t)$$

définie dans $\widetilde{V}$ peut s'écrire

$$\Delta(t) = \sum_{i=0}^{m} \delta_i N_i(t)$$

dans l'espace V en insérant tous les nœuds de $\mathbf{t} \setminus \widetilde{\mathbf{t}}$. L'insertion de nœuds consistant en des

combinaisons linéaires, le passage de $\widetilde{\delta} = \begin{bmatrix} \widetilde{\delta}_0 \\ \vdots \\ \widetilde{\delta}_m \end{bmatrix}$ à $\delta = \begin{bmatrix} \delta_0 \\ \vdots \\ \delta_m \end{bmatrix}$ peut s'écrire $\delta = P\widetilde{\delta}$, où P est une matrice de taille $m \times \widetilde{m}$ dépendant de $\mathbf{t}$. L'opération de déformation $\bar{\mathbf{c}} = \mathbf{c} + \Delta$ se concrétise donc par l'addition $\bar{\mathbf{c}}_i = \mathbf{c}_i + \delta_i$ des coefficients dans la base de V.

Imaginons par exemple que nous souhaitions déformer $\mathbf{c}$ de façon à ce que le point $\mathbf{c}(\bar{t})$ au paramètre $\bar{t}$ sur la courbe ait comme nouvelle position $\bar{\mathbf{c}}(\bar{t}) = \mathbf{c}(\bar{t}) + \vec{d}$. La déformation Δ_{pos} correspondante doit vérifier $\Delta_{pos}(\bar{t}) = \vec{d}$. La solution proposée est

$$\widetilde{\delta}_j = \frac{1}{\sigma} \widetilde{N}_j(\bar{t}) \vec{d} \quad \text{pour } j \in \{0, \ldots, \widetilde{m}\}$$

où

$$\sigma = \sum (\widetilde{N}_j(\bar{t}))^2 \ .$$

L'intérêt de cette définition est qu'elle prend en considération (par δ_i) les coefficients $\mathbf{c}_i$ proportionnellement à l'influence qu'ils ont sur la position du point $\mathbf{c}(\bar{t})$. C'est donc une déformation locale, mais dont l'étendue dépend des nœuds supprimés entre $\widetilde{\mathbf{t}}$ et $\mathbf{t}$. Si beaucoup de nœuds ont été supprimés la déformation est large, si peu de nœuds ont été supprimés la déformation est plus locale. C'est en ce sens que nous parlons de déformation *multi-échelle*.

Conservation de l'aire d'une courbe fermée.

Dans le cas où $\mathbf{c}(t)$ est fermée, l'aire incluse dans la courbe est, d'après le théorème de Green [EL91] :

$$\mathcal{A} = \frac{1}{2} \oint x(t)y'(t) - x'(t)y(t)dt.$$

Dans la base B-spline, l'aire s'écrit sous forme bilinéaire $2\mathcal{A} = X^T M Y$ avec les vecteurs coordonnées

$$X = \begin{bmatrix} x_0 \\ \vdots \\ x_m \end{bmatrix} \quad \text{et} \quad Y = \begin{bmatrix} y_0 \\ \vdots \\ y_m \end{bmatrix}$$

et la matrice $M = \left(m_{ij}\right)_{i,j}$ telle que

$$m_{ij} = \oint N_i(t)N_j'(t) - N_i'(t)N_j(t)\ dt.$$

Plaçons nous maintenant dans un processus de déformation. La courbe initiale $\mathbf{c}$ d'aire $\mathcal{A}_{ref}$ est déformée en $\bar{\mathbf{c}} = \mathbf{c}_{ref} + \Delta$ d'aire $\bar{\mathcal{A}}$. Ce processus de déformation intègre les contraintes linéaires (position, tangence, symétries) grâce à une minimisation : la déformation δ est minimisée au sens L_2 sous les contraintes linéaires.
Par exemple l'édition $\bar{\mathbf{c}}(\bar{t}) = \mathbf{c}_{ref}(\bar{t}) + \vec{d}$ précédemment citée se traduit par la contrainte $\Delta(\bar{t}) = \vec{d}$.

Souhaitant conserver l'aire, la contrainte $\bar{\mathcal{A}} = \mathcal{A}_{ref}$ est ajoutée. Cette contrainte s'écrit bilinéairement comme

$$X^T M Y = (X + \delta X)^T M (Y + \delta Y)$$

où δX et δY sont les vecteurs coordonnées de δ. Cette expression étant quadratique, il se pose le même problème de linéarisation que nous avons résolu section 2.4.3 par un développement limité au premier ordre. Une autre solution est proposée dans l'article [Elb01].
Supposons que δY est connu. La contrainte se réécrit :

$$\delta X^T M(Y + \delta Y) + X^T M \delta Y = 0$$

dans laquelle seul δX est inconnu. La contrainte est devenue linéaire et peut être traitée comme les contraintes de position ou de tangence. En pratique, les rôles de X et Y peuvent et doivent être interchangés, c'est-à-dire qu'une succession de petites déformations considéreront alternativement X ou Y comme fixe, ne calculant respectivement que δY ou δX.

Seulement, alors que les courbes $\mathbf{c}_{ref}$ et $\bar{\mathbf{c}}$ sont définies dans V, la déformation Δ est définie dans $\widetilde{V}$ par $\widetilde{\delta}$. En utilisant la matrice P d'insertion de nœuds, la contrainte pour X variable s'écrit :

$$\widetilde{\delta X}^T P^T M \ (Y + P\widetilde{\delta Y}) + X^T M \ P \ \widetilde{\delta Y} = 0$$

linéaire en $\widetilde{\delta X}$ puisque $\widetilde{\delta Y}$ est supposé connu.

2.5.2 Comparaison

Le cadre nommé *multi-échelle* dans la section 2.5.1 n'est pas exactement un cadre *multi-résolution* comme nous l'avons défini dans 1.3 et utilisé dans 2.4. Le parallèle peut être fait en assimilant $V = V^n$ et $\widetilde{V} = V^e$. La matrice d'insertion de nœuds joue le rôle d'une matrice de subdivision $P = P^n P^{n-1} \dots P^{e+1}$. La différence entre δ et $\widetilde{\delta}$ est alors la même qu'entre $\mathbf{c}^n$ et $\mathbf{c}^e$: l'un est l'expression en base fine de l'autre (défini en base grossière). En corollaire, on peut assimiler les fonctions de base aux fonctions d'échelle $N_i = \varphi_i^n$ et $\widetilde{N}_i = \varphi_i^e$.

La différence est à chercher dans les coefficients de détails $\mathbf{d}$ du modèle multirésolution qui simplement *n'existent pas* dans le modèle multi-échelle. Autrement dit ce dernier permet de raffiner une fonction grossière dans une base plus fine, mais il ne permet pas d'analyser un signal fin à différentes échelles comme par le banc de filtres, FIG. 1.1. Ainsi, nous référant à la notion de boucle d'édition introduite dans la section 1.4, on constate que la méthode multi-échelle ne suit réellement aucune des boucles d'édition de la figure 1.5 page 10. En effet, la courbe n'est jamais décomposée à une échelle grossière. Seule la déformation est construite dans l'espace d'approximation $\widetilde{V}$, puis raffinée pour être additionnée à la courbe.

Nous nous souviendrons alors qu'un argument fort de la méthode multirésolution (voir section 2.4.4) est de pouvoir fixer les coefficients de détails pour ne modifier que les coefficients d'échelle. Dans ce cas les deux méthodes atteignent exactement le même objectif : déformer la courbe à une échelle large et contrôlable tout en conservant les détails et l'aire.

Pour mieux comprendre les avantages de notre méthode multirésolution par rapport à la méthode multi-échelle de [Elb01] il faut s'extraire du cadre "animation" dans lequel nous illustrons ces travaux. Le calcul d'aire que nous avons présenté dans la section 2.3 peut être réutilisé dans n'importe quelle application qui manipule et modifie des courbes multirésolution. Nous avons souligné dans la section 1.4 que l'étape d'édition, comme nous la nommons en FIG. 1.5, peut avoir n'importe quelle origine. Si cette étape modifie des coefficients de détails, une analyse multirésolution s'avère nécessaire, et l'édition multi-échelle ne suffit plus.

Prenons l'exemple de la compression de données, qui fait fréquemment appel à des analyses en ondelettes. Imaginons que nous souhaitions compresser les données qui codent la courbe en utilisant une décomposition en ondelettes. La première étape consistera à faire un seuillage sur

les coefficients multirésolution à toutes les échelles. Ce seuillage sera exactement ce que nous avons appelé "édition" pour l'animation, en ce sens que le seuillage édite un certain nombre de coefficients (en les mettant à 0). Sa particularité, par rapport à l'édition d'un coefficient d'échelle $\mathbf{c}_i^e$ proposé section 2.4.1, est qu'il édite plusieurs coefficients d'échelle *et* de détails, nécessitant donc une analyse multirésolution sous la forme (2.11). Nous pouvons toujours appliquer la conservation d'aire en aval, à partir des valeurs X_0 et Y_0 issus du seuillage, de façon à compresser la courbe tout en conservant son aire.

Remarquons pour terminer que les différences entre les méthodes d'*intégration des contraintes*, comme nous les avons appelées dans l'introduction, ne sont pas discriminantes :

- L'énergie de tension utilisée dans notre méthode multirésolution est exactement adaptable à la méthode multi-échelle qui procède aussi par optimisation.
- Les méthodes de linéarisation (la fixation par alternance de X et de Y pour la méthode multi-échelle, contre l'approximation par développement limité pour la méthode multirésolution) sont interchangeables.

Ayant implémenté chacune des possibilités pour la méthode multirésolution, nous en avons tiré les conclusions suivantes :

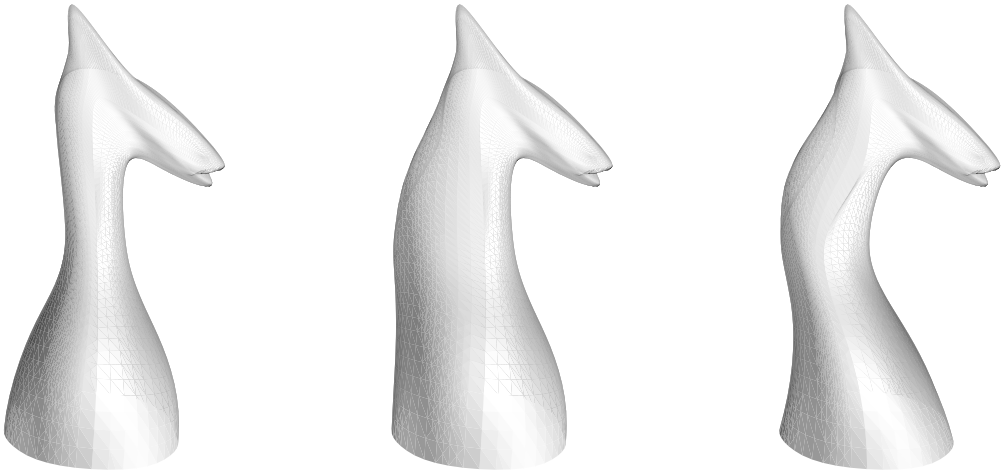
- Dans la plupart des utilisations la valeur de β est proche de 1, ce qui signifie que l'énergie de tension a peu de poids dans la minimisation. En pratique l'aspect lisse de la courbe est assuré par des B-splines de degré 2 ou 3. Bien qu'apportant plus de souplesse, l'énergie de tension est donc peu utile pour des applications en animation.
- Dans la section 2.4.3 nous avons brièvement justifié l'utilisation du développement limité parce qu'il ne brise pas la symétrie entre X et Y, comme le fait la linéarisation de [Elb01]. Dans un grand nombre de cas la différence visuelle est minime, mais il est des cas particuliers où la différence est non négligeable. Supposons que l'on se trouve dans le cas où Y est fixé, donc $(\bar{x}(t), \bar{y}(t)) = (x(t) + \Delta x(t), y(t))$. La différence d'aire s'écrit sous forme intégrale :

$$\bar{\mathcal{A}} - \mathcal{A}_{ref} = \oint \Delta x(t) y'(t) - \Delta x'(t) y(t) dt \ .$$

  Il s'en suit que si, sur la portion de courbe déformée (*i.e.* la portion sur laquelle $\Delta x(t) \neq 0$), la courbe est parallèle à l'axe des abscisses (*i.e.* $y' = 0$, *i.e.* y constant), on obtient $\bar{\mathcal{A}} - \mathcal{A}_{ref} = 0$. Ceci signifie que la déformation Δx ne peut en aucun cas corriger une variation d'aire. C'est évidemment un cas pathologique, mais qui dénote un risque d'instabilité notoire : si y est "presque constant", il faudra une grande déformation en X pour corriger un petit peu l'aire. En des termes plus mathématiques, il faut comprendre que le système linéaire issu de la minimisation peut être mal conditionné, et donc être sujet à des instabilités numériques. C'est pourquoi la solution du développement limité nous semble plus satisfaisante.

Chapitre 3

Volume et surfaces B-spline



Nous présentons dans ce chapitre deux méthodes de déformation à volume constant de surfaces composites constituées de patchs B-spline produit tensoriel. La première investigue le cas des B-splines non uniformes et la deuxième le cas uniforme. Toutes deux sont basées sur une linéarisation du calcul du volume, suivie d'une minimisation sous contrainte. Elles se différencient principalement par le fait que la première propose un cadre de déformation multi-échelle d'une surface fine, alors que la seconde propose un cadre multirésolution complet et unitaire pour la surface et la déformation.

3.1 Problématique

Les surfaces B-splines produit tensoriel, uniformes ou non uniformes, sont des modèles de surfaces très utilisés en CGAO et en informatique graphique car ils sont efficaces pour représenter et manipuler des surfaces lisses à partir de peu de points de contrôle. En outre, il existe des schémas éprouvés pour représenter et éditer ces surfaces dans des bases multirésolution. Nous verrons que la principale différence entre le modèle uniforme et le modèle non uniforme est qu'ils n'acceptent pas le même type de schémas multirésolution.

Nous proposons d'étudier ici la conservation du volume englobé par ces surfaces lors de déformations. Cette contrainte, comme la contrainte d'aire pour les courbes fermées, a pour but d'améliorer le réalisme des animations qui utilisent ces surfaces pour modéliser des objets mous. Ces travaux sont le fruit d'une collaboration initiée en 2004 avec Gershön Elber, du département d'informatique de Technion (Haifa, Israël).

Dans un souci d'exhaustivité, nous proposons d'étudier les surfaces uniformes et les surfaces non uniformes, qui sont l'extension produit tensoriel des courbes utilisées dans le chapitre précédent. Pour autant la conservation du volume n'est pas la généralisation immédiate aux surfaces de la conservation de l'aire des courbes. En effet, une courbe B-spline *fermée* s'obtient facilement par périodisation de la séquence de nœuds sur l'espace de paramétrisation $\mathbb{R}$. Les points de de contrôle sont alors les coefficients dans une base périodisée de $\mathbb{R}$. La périodisation des séquences de nœuds en produit tensoriel conduit à ne traiter que des surfaces à topologie torique. Il est bien évidemment trop restrictif de limiter notre étude aux tores. De plus, les surfaces complexes ne sont souvent pas définies par une seule surface B-spline, mais par une collection de surfaces, appelées *patchs* ou *carreaux*, qui se recollent entre eux par leurs courbes de bord. Nous allons donc baser nos études sur les patchs, avant de les généraliser aux surfaces composites.

Nous apportons une attention particulière à la complexité des algorithmes que nous développons. Afin de permettre une édition interactive de surfaces sous contrainte de volume, nous essayons de faire le plus de traitements possibles *avant* la déformation elle-même, *i.e.* un maximum de pré-calculs.

Ce chapitre s'articule autour des deux modèles de surfaces, non uniformes (3.2) puis uniformes (3.3). Pour chacun d'eux, nous commençons par développer une méthode de calcul du volume, applicable quel que soit le niveau de résolution auquel la surface est représentée. Ensuite nous exposons des méthodes de déformation d'un patch à volume constant, basées sur une minimisation sous contrainte. Les minimisations se résolvent par un système linéaire creux. Nous exhibons la solution *explicite* de ces systèmes, ce qui est très important pour l'efficacité de nos méthodes. Ces méthodes sont ensuite étendues aux surfaces composites, avant d'être illustrées et commentées.

Dans une dernière section (3.4), nous comparons les deux modèles, afin de clarifier leurs points communs et leurs différences. Nous discutons aussi des conséquences de ces différences, du point de vue de l'utilisateur.

3.2 B-splines non uniformes

Les surfaces que nous utilisons ici sont la généralisation par produit tensoriel des B-splines non uniformes présentées en section 1.2 et utilisées en 2.5. Comme nous allons chercher à conserver le volume englobé par cette surface, il est *a priori* nécessaire qu'elle soit fermée, c'est-à-dire

sans bords. De plus elle doit être orientable, *i.e.* on doit pouvoir définir un intérieur et un extérieur.

Nous allons mener cette étude sur un unique patch dans un premier temps, puis sur les surfaces composites. L'étude sur un patch unique est non seulement une première étape vers une solution pour les surfaces composites, mais en outre nous montrons qu'il n'est pas absurde de parler de conservation de volume pour un patch, bien qu'il ne modélise pas une surface fermée.

Dans la section 3.2.1 nous présentons le modèle de surfaces et le type des déformations multi-échelles qui s'y appliquent. Ensuite la méthode de calcul du volume est expliquée en 3.2.2. Deux méthodes de déformation de patchs à volume constant sont détaillées dans les sections 3.2.3 et 3.2.4 : la première consiste à éditer la surface avant de la corriger pour assurer la contrainte de volume, alors que la deuxième traite simultanément l'édition et la contrainte. Ces deux méthodes sont étendues aux surfaces composites dans les sections 3.2.6 et 3.2.7, après avoir abordé quelques difficultés spécifiques aux surfaces composites dans la section 3.2.5. La section 3.2.8 regroupe des illustrations commentées et une comparaison plus poussée des deux méthodes.
Un récapitulatif des notations utilisées dans ces sections est donné page 41.

3.2.1 Surfaces produit tensoriel

Une surface composite S est définie par une collection de patchs produit tensoriel $\{S^q\}_q$, définis chacun sur leur propre espace de paramétrisation (un pavé de $\mathbb{R}^2$). Ces patchs se recollent selon leurs courbes de bords, qui sont l'image d'un bord du pavé de paramétrisation. Dans la présente section nous définissons ces patchs et les déformations qui s'y appliquent.

Patchs produit tensoriel.

Pour définir le produit tensoriel sur un pavé de $\mathbb{R}^2$, nous avons besoin de deux séquences de nœuds, sur lesquelles nous allons définir deux séries de B-splines unidimensionnelles à multiplier l'une par l'autre. Soient

$$\mathbf{u} = \{u_0, u_1, \dots, u_{m_u+d}\}$$
$$\text{et} \quad \mathbf{v} = \{v_0, v_1, \dots, v_{m_v+d}\}$$

deux séquences de nœuds non uniformes, croissantes, dans $\mathbb{R}$. Pour simplifier les notations nous supposerons, sans restriction, que $u_0 = v_0 = 0$ et $u_{m_u+d} = v_{m_v+d} = 1$. D'après les équations de la section 1.2, nous pouvons définir sur ces séquences deux familles de fonctions B-spline $\left(N^d_{i_u}(u)\right)_{0\le i_u<m_u}$ et $\left(N^d_{i_v}(v)\right)_{0\le i_v<m_v}$ de degré d.

Remarque : le degré d peut être différent en u et en v, et c'est souvent le cas en pratique. Pour alléger les notations nous supposons que c'est le même, et nous l'omettons quand il ne joue aucun rôle dans les calculs. Tous les résultats présentés ont été implémentés avec des degrés différents (l'un pour $N_{i_u}(u)$, l'autre pour $N_{i_v}(v)$), mais cela ne change pas les méthodes que nous allons présenter, puisque la particularité du produit tensoriel est justement de séparer les deux coordonnées.

À partir de là nous définissons une base de fonctions B-splines produit tensoriel :

$$N_{\mathbf{i}}(u,v) = N_{i_u}(u)N_{i_v}(v) \quad \text{pour } (0,0) \le \mathbf{i} = (i_u, i_v) \le (m_u - 1, m_v - 1)\ . \tag{3.1}$$

Ces fonctions constituent une base de l'espace V, qui est un espace de fonctions polynomiales (à deux variables) de degré d par morceaux (sur les pavés $[u_{i_u}, u_{i_u+1}]\times[v_{i_v}, v_{i_v+1}]$, pour les indices

$i_u \in \{d, \ldots, m_u - 1\}$, $i_v \in \{d, \ldots, m_v - 1\}$), avec une continuité d'ordre $d-1$ en tous les nœuds intérieurs (à la condition qu'ils soient distincts). Un *patch* produit tensoriel dans V est défini comme l'application

$$S^q \ : \ [0,1]^2 \longrightarrow \mathbb{R}^3 \tag{3.2}$$

$$(u,v) \longrightarrow S^q(u,v) = \sum_{\mathbf{i}=(0,0)}^{(m_u-1,m_v-1)} \mathbf{p_i}\ N_{\mathbf{i}}(u,v) \tag{3.3}$$

où $\mathbf{p_i} = (x_{\mathbf{i}}, y_{\mathbf{i}}, z_{\mathbf{i}}) \in \mathbb{R}^3$ sont appelés *points de contrôle*. Sur la figure 3.1 les points de contrôle sont les sommets du maillage quadrangulaire et $S^q(u,v)$ est la surface lisse.

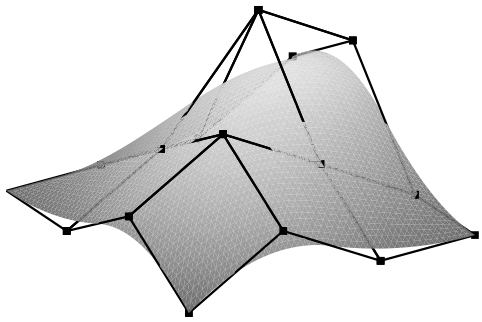


FIG. 3.1 – Patch B-spline et son maillage de contrôle

Propriétés de la base B-spline produit tensoriel.

La base de fonctions $N_{\mathbf{i}}$ hérite d'un certain nombre de propriétés des bases unidimensionnelles, et en particulier le support compact :

$$\text{Support}\big(N_{\mathbf{i}}(u,v)\big) = [u_{i_u}; u_{i_u+d+1}] \times [v_{i_v}; v_{i_v+d+1}] \quad \text{pour } (0,0) \leq \mathbf{i} \leq (m_u - 1, m_v - 1)\ .$$

La plupart des formules présentées en annexe A.1 sont applicables en produit tensoriel, c'est-à-dire sous forme de convolution unidirectionnelle sur les lignes puis sur les colonnes de points de contrôle. Entre autres les formules sur les dérivées (partielles), l'insertion de nœuds et l'algorithme de De Boor (pour calculer la position d'un point sur la surface) nous seront utiles.

Conditions d'interpolation au bord dans la définition d'un patch S^q.

Pour compléter la définition (3.3), nous imposons les conditions suivantes sur les nœuds de bord :

$$u_0 = u_1 = \ldots = u_d \ = \ 0 \ = \ v_0 = \ldots = v_d$$

et

$$u_{m_u} = u_{m_u+1} = \ldots = u_{m_u+d} \ = \ 1 \ = \ v_{m_v} = \ldots = v_{m_v+d}.$$

Il s'agit de conditions d'interpolation aux bords, c'est-à-dire que les *courbes de bord* sont des courbes B-splines entièrement définies par les points de bord :

$$\begin{aligned} S^q(0,v) &= \sum_{i_v} \mathbf{p}_{0,i_v}\ N^d_{i_v}(v)\ , \\ S^q(1,v) &= \sum_{i_v} \mathbf{p}_{m_u-1,i_v}\ N^d_{i_v}(v)\ , \\ S^q(u,0) &= \sum_{i_u} \mathbf{p}_{i_u,0}\ N^d_{i_u}(u)\ , \\ S^q(u,1) &= \sum_{i_u} \mathbf{p}_{i_u,m_v-1}\ N^d_{i_u}(u)\ , \end{aligned}$$

et par conséquent les points de contrôle aux angles sont interpolés :

$$S^q(0,0) = \mathbf{p}_{0,0}\ ,\ S^q(1,0) = \mathbf{p}_{m_u-1,0}\ ,\ S^q(1,1) = \mathbf{p}_{m_u-1,m_v-1} \text{ et } S^q(0,1) = \mathbf{p}_{0,m_v-1}\ .$$

Déformation multi-échelle.

Nous souhaitons maintenant déformer le patch $S^q \in V$ à une échelle grossière. Pour cela nous définissons un espace d'approximation $\widetilde{V} \subset V$ à partir de deux sous-séquences de nœuds $\widetilde{\mathbf{u}} = \{\widetilde{u}_0, \ldots, \widetilde{u}_{\widetilde{m}_u+d}\} \subset \mathbf{u}$ et $\widetilde{\mathbf{v}} = \{\widetilde{v}_0, \ldots, \widetilde{v}_{\widetilde{m}_v+d}\} \subset \mathbf{v}$ tels que :

– Les nœuds de bord sont conservés, *i.e* pour $i \in \{0, 1, \ldots, d\}$:

$$\widetilde{u}_i = u_i \quad \text{et} \quad \widetilde{u}_{\widetilde{m}_u+i} = u_{m_u+i}$$

$$\widetilde{v}_i = v_i \quad \text{et} \quad \widetilde{v}_{\widetilde{m}_v+i} = v_{m_v+i}\ .$$

– Une partie des nœuds intérieurs est supprimée :

$$\begin{aligned} \{\widetilde{u}_{d+1}, \ldots, \widetilde{u}_{\widetilde{m}_u-1}\} &\subsetneq \{u_{d+1}, \ldots, u_{m_u-1}\} \\ \{\widetilde{v}_{d+1}, \ldots, \widetilde{v}_{\widetilde{m}_v-1}\} &\subsetneq \{v_{d+1}, \ldots, v_{m_v-1}\}\ . \end{aligned}$$

On introduit (comme Eq. (3.1)) une base dite grossière de fonctions B-splines :

$$\widetilde{N}_{\mathbf{j}}(u,v) = \widetilde{N}^d_{j_u}(u)\widetilde{N}^d_{j_v}(v) \quad \text{pour} \quad (0,0) \leq \mathbf{j} = (j_u, j_v) \leq (\widetilde{m}_u - 1, \widetilde{m}_v - 1)$$

Une déformation $\Delta(u,v)$ est alors définie comme élément de l'espace $\widetilde{V}$ engendré par les fonctions $\widetilde{N}_{\mathbf{j}}$:

$$\Delta(u,v) = \sum_{\mathbf{j}} \widetilde{\delta}_{\mathbf{j}} \widetilde{N}_{\mathbf{j}}(u,v) \quad \text{avec} \quad \widetilde{\delta}_{\mathbf{j}} \in \mathbb{R}^3\ . \tag{3.4}$$

Pour appliquer la déformation à un patch S^q, c'est-à-dire calculer $S^q + \Delta$, il faut exprimer Δ dans l'espace V :

$$\Delta(u,v) = \sum_{\mathbf{i}} \delta_{\mathbf{i}} N_{\mathbf{i}}(u,v) \quad \text{avec} \quad \delta_{\mathbf{i}} \in \mathbb{R}^3,\ (0,0) \leq \mathbf{i} \leq (m_u - 1, m_v - 1)$$

Ceci est possible grâce à l'insertion de nœuds [Far01], qui permet de calculer les $\delta_{\mathbf{i}}$ à partir des $\widetilde{\delta}_{\mathbf{j}}$. La formule unidimensionnelle d'insertion de nœuds est rappelée en annexe A.1. Dans le formalisme produit tensoriel, on travaille sur les colonnes puis sur les lignes de $\widetilde{\delta}$:

pour $\hat{u} \in \mathbf{u} \setminus \widetilde{\mathbf{u}}$ **faire**
  pour $j_v \in \{0, \ldots, \widetilde{m}_v - 1\}$ **faire**
    inserer_noeud($\hat{u}$) sur la colonne j_v
  fin pour
fin pour
pour $\hat{v} \in \mathbf{v} \setminus \widetilde{\mathbf{v}}$ **faire**
  pour $j_u \in \{0, \ldots, m_u - 1\}$ **faire**
    inserer_noeud($\hat{v}$) sur la ligne j_u
  fin pour
fin pour

Déplacement d'un point sur la surface.

On considère la déformation Δ_{pos} particulière consistant à déplacer un point du patch. Pour éditer la surface, nous allons manipuler directement des points sur la surface. Formalisons-le par un vecteur déplacement $\vec{d}$ pour un point de coordonnées $(\bar{u}, \bar{v})$. Nous avons à définir une déformation $\Delta_{pos} \in \widetilde{V}$ telle que :

$$S^q(\bar{u}, \bar{v}) + \Delta_{pos}(\bar{u}, \bar{v}) = S^q(\bar{u}, \bar{v}) + \vec{d} \, ,$$

c'est-à-dire telle que $\Delta_{pos}(\bar{u}, \bar{v}) = \vec{d}$. Logiquement, on souhaite que la déformation soit maximale en $(\bar{u}, \bar{v})$. L'étendue de la déformation, quant à elle, doit dépendre de l'espace d'approximation $\widetilde{V}$, *i.e.* des sous-séquences de nœuds $\widetilde{\mathbf{u}}$ et $\widetilde{\mathbf{v}}$. Généralisant la solution proposée dans [Elb01] et rappelée dans la section 2.5, nous choisissons :

$$\widetilde{\delta}_{\mathbf{j}} = \frac{1}{\sigma} \, \widetilde{N}_{\mathbf{j}}(\bar{u}, \bar{v}) \, \vec{d} \quad \text{pour } \mathbf{j} \in \{0, \ldots, \widetilde{m}_u\} \times \{0, \ldots, \widetilde{m}_v\} \, ,$$

où

$$\sigma = \sum (\widetilde{N}_{\mathbf{j}}(\bar{u}, \bar{v}))^2 \, .$$

Pour calculer les coefficients $\widetilde{\delta}_{\mathbf{j}}$ on pourra utiliser l'algorithme de De Boor [Far01].

L'étendue de la déformation dépend des sous-séquences de nœuds. Comme chaque B-spline $\widetilde{N}_{j_u, j_v}$ a un support compact $[\widetilde{u}_{j_u}, \widetilde{u}_{j_u+d+1}] \times [\widetilde{v}_{j_v}, \widetilde{v}_{j_v+d+1}]$, le coefficient $\widetilde{\delta}_{\mathbf{j}}$ est non nul si et seulement si $(\bar{u}, \bar{v})$ appartient au support. Posons $\bar{j_u}$ et $\bar{j_v}$ les indices des nœuds encadrant $\bar{u}$ et $\bar{v}$, c'est-à-dire tels que

$$\widetilde{u}_{\bar{j_u}} \leq \bar{u} < \widetilde{u}_{\bar{j_u}+1}$$
$$\text{et} \quad \widetilde{v}_{\bar{j_v}} \leq \bar{v} < \widetilde{v}_{\bar{j_v}+1} \, .$$

Alors

$$\widetilde{\delta}_{\mathbf{j}} \neq 0 \quad \Leftrightarrow \quad (\bar{j_u} - d, \bar{j_v} - d) \leq \mathbf{j} \leq (\bar{j_u}, \bar{j_v}) \, ,$$

donc le support de Δ_{pos} est

$$[\widetilde{u}_{\bar{j_u}-d}, \widetilde{u}_{\bar{j_u}+d+1}] \times [\widetilde{v}_{\bar{j_v}-d}, \widetilde{v}_{\bar{j_v}+d+1}] \, .$$

Ce support est l'*étendue* de la déformation, qui est donc bien contrôlée par les sous-séquences de nœuds : si beaucoup de nœuds ont été supprimés la déformation est large, et si peu de nœuds ont été supprimés elle est plus localisée.

La figure 3.2 illustre le rôle des sous-séquences de nœuds $\widetilde{\mathbf{u}} \subset \mathbf{u}$ et $\widetilde{\mathbf{v}} \subset \mathbf{v}$. Un patch initialement plan est déformé deux fois avec les mêmes valeurs de $(\bar{u}, \bar{v})$ et $\vec{d}$ mais pour des sous-séquences différentes. Au milieu tous les nœuds sont conservés et la déformation est localisée, alors qu'à droite, où seulement un nœud intérieur sur trois a été conservé, la déformation est plus large.

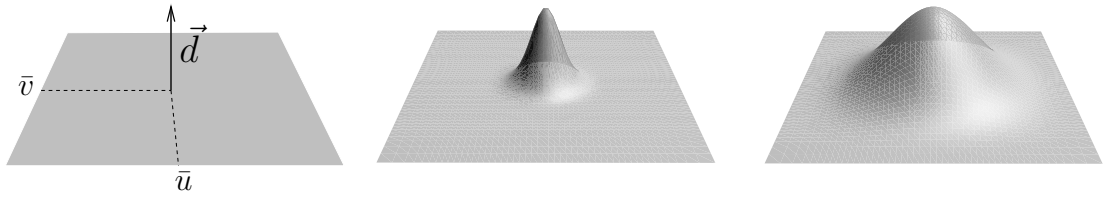


Fig. 3.2 – Étendue de la déformation.
Un patch plan (à gauche) est déformé pour différentes sous-séquences de nœuds. Au milieu : tous les nœuds sont conservés ($\widetilde{V} = V$). À droite : 2 nœuds intérieurs sur 3 ont été supprimés.

3.2.2 Calcul du volume

Notre objectif est d'appliquer une déformation $\Delta \in \widetilde{V}$ à un patch $S^q \in V$ tout en conservant le volume, noté Θ. Par conséquent, il nous faut établir une méthode de calcul de volume non seulement pour S^q (défini dans la base $(N_\mathbf{i})_\mathbf{i}$ de V), mais également pour $S^q + \Delta$ sachant que Δ est définie dans la base $(\widetilde{N}_\mathbf{j})_\mathbf{j}$. Nous allons donc présenter d'abord le calcul de volume en base fine (pour S^q), puis le calcul de volume avec les deux bases (pour $S^q + \Delta$).

Volume de la surface initiale.

Le volume inclus dans une surface fermée peut s'exprimer sous la forme d'une intégrale sur la surface [GOMP98, Elb01]. Dans le cas d'une surface composite, le volume est la somme des intégrales selon chaque patch :

$$\Theta(S) = \sum_{\text{patchs } q} \Theta(S^q)$$

avec

$$\Theta(S^q) = \int_{v=0}^{1}\int_{u=0}^{1} z(u,v)\Big(\frac{\partial x}{\partial u}(u,v)\frac{\partial y}{\partial v}(u,v) - \frac{\partial x}{\partial v}(u,v)\frac{\partial y}{\partial u}(u,v)\Big)\,du\,dv \tag{3.5}$$

où $S^q(u,v) = (x(u,v), y(u,v), z(u,v))$ est défini par (3.3). Pour que ce calcul soit valide il faut que la surface soit orientée, et plus strictement que tous les patchs qui la composent soient orientés identiquement. Le volume ainsi calculé est un volume signé : si la normale à la surface $\frac{\partial S}{\partial u} \wedge \frac{\partial S}{\partial v}$ est sortante le volume est positif, dans le cas contraire elle est négative.

L'intégrale (3.5), que nous appelons "volume" du patch S^q, peut s'interpréter géométriquement comme le volume de la colonne entre le plan xy et le patch (voir Fig. 3.3). Cela est assez clair quand $x(u,v) = u$ et $y(u,v) = v$ car alors $z(x,y)$ devient une fonction hauteur et le volume de la "colonne" s'exprime comme

$$\iint z(x,y)\;dx\,dy\;.$$

Ces considérations qualitatives nous permettent de justifier la marche que nous allons suivre pour présenter ces travaux, à savoir que nous allons commencer par développer nos méthodes de déformation sur un seul patch avant d'aborder les surfaces composites. Même si le "volume"

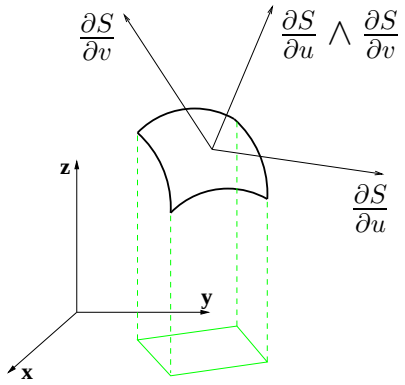


FIG. 3.3 – Élément de volume.
L'élément de surface (en noir) porte deux dérivées partielles et une normale. "L'élément projeté" (quadrilatère du plan xy) a pour aire élémentaire $(x_u y_v - x_v y_u)\, du\, dv$.

d'un patch correspond au volume de la "colonne" (FIG. 3.3), et non au volume englobé par une surface fermée, il est pertinent de le déformer à volume constant. En effet, considérant une surface composite que l'on déforme sur un seul patch, la *variation* de volume du patch *est* la variation de volume de la surface composite, du moment qu'on prend soin de fixer les bords du patch en question pour assurer la continuité de la surface.

En reportant l'expression (3.3) de S^q dans l'équation (3.5) de $\Theta(S^q)$, le volume s'exprime sous la forme tri-linéaire

$$\Theta(S^q) = \sum_{\mathbf{i},\mathbf{j},\mathbf{k}} \theta_{\mathbf{i}\,\mathbf{j}\,\mathbf{k}} x_{\mathbf{i}}\, y_{\mathbf{j}}\, z_{\mathbf{k}} \quad \text{avec} \quad (0,0) \leq \mathbf{i},\mathbf{j},\mathbf{k} \leq (m_u - 1, m_v - 1) \tag{3.6}$$

où

$$\theta_{\mathbf{i}\,\mathbf{j}\,\mathbf{k}} \;=\; \iint N_{\mathbf{k}} \left(\frac{\partial N_{\mathbf{i}}}{\partial u}\frac{\partial N_{\mathbf{j}}}{\partial v} - \frac{\partial N_{\mathbf{i}}}{\partial v}\frac{\partial N_{\mathbf{j}}}{\partial u}\right) \quad \in \mathbb{R}.$$

et $(x_{\mathbf{i}}, y_{\mathbf{i}}, z_{\mathbf{i}}) = \mathbf{p_i} \in \mathbb{R}^3$ sont les points de contrôle. Pour calculer ces coefficients $\theta_{\mathbf{i}\,\mathbf{j}\,\mathbf{k}}$, il est utile de séparer les intégrales en u et en v :

$$\begin{aligned}\theta_{\mathbf{i}\,\mathbf{j}\,\mathbf{k}} \;&=\; \int_0^1 N'_{i_u}(u) N_{j_u}(u) N_{k_u}(u)\, du \int_0^1 N_{i_v}(v) N'_{j_v}(v) N_{k_v}(v)\, dv \\ &\quad - \int_0^1 N_{i_u}(u) N'_{j_u}(u) N_{k_u}(u)\, du \int_0^1 N'_{i_v}(v) N_{j_v}(v) N_{k_v}(v)\, dv \\ &=\; \theta^u_{j_u k_u i_u}\theta^v_{i_v k_v j_v} - \theta^u_{i_u k_u j_u}\theta^v_{j_v k_v i_v} \end{aligned} \tag{3.7}$$

avec

$$\theta^u_{ijk} = \theta^u_{jik} = \int_0^1 N_{i_u}(u)\, N_{j_u}(u)\, N'_{k_u}(u)\, du \tag{3.8}$$

$$\text{et} \quad \theta^v_{ijk} = \theta^v_{jik} = \int_0^1 N_{i_v}(v)\, N_{j_v}(v)\, N'_{k_v}(v)\, dv\ . \tag{3.9}$$

Le calcul de ces derniers coefficients peut se faire par intégration formelle dans le cas des B-splines, mais pourrait nécessiter l'utilisation de formules de quadrature dans un cadre plus

général. Nous avons opté pour une solution hybride, qui consiste à calculer numériquement les coefficients des polynômes sur chaque intervalle, puis à les intégrer formellement.

Remarque : le calcul du volume nous a posé quelques problèmes de précision lorsque le degré des B-splines augmente (à partir du degré 5). Quoique mineurs, ces problèmes sont à l'heure actuelle non encore élucidés. Nous envisageons de mettre en œuvre un calcul totalement formel pour nous mieux contrôler la précision des coefficients θ^u_{ijk} et θ^v_{ijk}.

Récapitulatif des notations.

Nous récapitulons ici les principales notations de la section 3.2. Certaines ne sont pas encore définies, mais le lecteur pourra ainsi se reporter à ce récapitulatif en cas de besoin.

Coordonnées. Dans les sections 3.2.3 et 3.2.4 nous manipulons les surfaces axe par axe, ce qui nous incite à utiliser les notations vectorielles :

$$X = \big(x_{\mathbf{i}}\big)_{\mathbf{i}} \in \mathbb{R}^{m_u m_v},\ Y = \big(y_{\mathbf{i}}\big)_{\mathbf{i}} \in \mathbb{R}^{m_u m_v} \text{ et } Z = \big(z_{\mathbf{i}}\big)_{\mathbf{i}} \in \mathbb{R}^{m_u m_v}\ .$$

où $(x_{\mathbf{i}}, y_{\mathbf{i}}, z_{\mathbf{i}}) = \mathbf{p_i} \in \mathbb{R}^3$ sont les points de contrôle.

Base fine est base grossière. Toutes les variables, tous les coefficients et toutes les fonctions ornées d'un tilde "˜" sont en base grossière, les autres sont en base fine.

Volume. Notons $\Theta_{YZ} = \big(\Theta_{YZ}(\mathbf{i})\big)_{\mathbf{i}} \in \mathbb{R}^{m_u m_v}$ le vecteur des

$$\Theta_{YZ}(\mathbf{i}) = \sum_{\mathbf{j},\mathbf{k}} \theta_{\mathbf{ijk}} y_{\mathbf{j}}\, z_{\mathbf{k}} \quad \in \mathbb{R} \quad \text{pour} \quad (0,0) \le \mathbf{i} \le (m_u - 1, m_v - 1) \tag{3.10}$$

ainsi que Θ_{XZ} et Θ_{XY} définis symétriquement.
Nous utilisons aussi les sommes partielles (définies Eq. (3.13) page 43)

$$\widetilde{\Theta}_{YZ}(\mathbf{i}) \in \mathbb{R},\ \widetilde{\Theta}_{XZ}(\mathbf{i}) \in \mathbb{R} \text{ et } \widetilde{\Theta}_{XY}(\mathbf{i}) \in \mathbb{R} \quad \text{pour } (0,0) \le \mathbf{i} \le (\widetilde{m}_u - 1, \widetilde{m}_v - 1)\ .$$

Elles sont regroupées dans les vecteurs $\widetilde{\Theta}_{YZ}$, $\widetilde{\Theta}_{XZ}$ et $\widetilde{\Theta}_{XY}$ de $\mathbb{R}^{\widetilde{m}_u \widetilde{m}_v}$.

Produits scalaires. Nous distinguons les produits scalaires dans $\mathbb{R}^3$ notés par le point "$\cdot$", et les produits scalaires dans $\mathbb{R}^{m_u m_v}$ ou $\mathbb{R}^{\widetilde{m}_u \widetilde{m}_v}$ notés $\langle\ ,\ \rangle$. Ceci nous permet de réécrire le volume (3.6) sous forme de produits scalaires en isolant chaque axe :

$$\Theta(S^q) = \langle\ \Theta_{XY}\ ,\ Z\ \rangle = \langle\ \Theta_{XZ}\ ,\ Y\ \rangle = \langle\ \Theta_{YZ}\ ,\ X\ \rangle\ . \tag{3.11}$$

Déformations. Les déformation $\Delta \in \widetilde{V}$ sont définies par $\widetilde{m}_u \widetilde{m}_v$ coefficients $\widetilde{\delta}_{\mathbf{i}} = (\widetilde{\delta} x_{\mathbf{i}}, \widetilde{\delta} y_{\mathbf{i}}, \widetilde{\delta} z_{\mathbf{i}})$ dans $\mathbb{R}^3$. Selon les besoins nous regroupons les coefficients pour chacun des axes dans un vecteur de $\mathbb{R}^{\widetilde{m}_u \widetilde{m}_v}$:

$$\widetilde{\delta} X = \big(\widetilde{\delta} x_{\mathbf{i}}\big)_{\mathbf{i}}\ , \widetilde{\delta} Y = \big(\widetilde{\delta} y_{\mathbf{i}}\big)_{\mathbf{i}} \text{ et } \widetilde{\delta} Z = \big(\widetilde{\delta} z_{\mathbf{i}}\big)_{\mathbf{i}}\ .$$

Aspects algorithmiques.

Il est impossible de calculer les intégrales 3.8 et 3.9 à la volée chaque fois qu'elles interviennent dans le processus, pour des raisons évidentes d'efficacité. Du point de vue du temps de calcul, le plus efficace serait de stocker tous les coefficients $\theta_{\mathbf{ijk}}$ (pré-calculés une fois pour toutes), mais ils sont au nombre $m_u^3 m_v^3$. Il est donc impensable de les stocker ainsi, car m_u et m_v

sont typiquement de l'ordre de quelques dizaines. La formulation (3.7) offre un bon compromis entre place mémoire et temps de calcul : les coefficients θ^u_{ijk} et θ^v_{ijk} sont pré-calculés et stockés pour une place $m_u^3 + m_v^3$, ce qui est acceptable. En contrepartie, le calcul du volume par les formules (3.6) et (3.7) se fait en temps $O(m_u^3 m_v^3)$.

Il est possible d'améliorer ces performances en prenant en compte le support compact des B-splines. Les intégrales (3.8) et (3.9) sont nulles dès que les supports de deux des fonctions intégrées (N_i, N_j et N'_k) sont disjoints, c'est-à-dire dès que $|i-j|$, $|i-k|$ ou $|j-k|$ dépasse d. Par conséquent il suffit de pré-calculer et de stocker les θ^u_{ijk} et θ^v_{ijk} pour $|i-j| \leq d$, $|i-k| \leq d$ et $|j-k| \leq d$.

Au niveau de l'implémentation, nous avons construit pour chaque valeur de $k = 0, 1, \ldots, m_u$ un tableau θ^u_k de taille $(2d+1)^2$ qui contient les valeurs θ^u_{ijk} pour $k-d \leq i, j \leq k+d$. De la même façon sont construits θ^v_k pour la seconde coordonnée.

La place mémoire utilisée n'est plus que de $(2d+1)^2(m_u + m_v)$. Parallèlement le calcul du volume par (3.6) diminue jusqu'en $O(d^4 m_u m_v)$ car la somme n'est effectuée que pour un petit nombre d'indices. Le calcul est ainsi linéaire par rapport au nombre de points de contrôle $(m_u m_v)$, donc il est asymptotiquement optimal.

Nous remarquons qu'il devient envisageable de stocker toutes les valeurs $\theta_{\mathbf{ijk}}$ non nulles, puisqu'ils ne sont que $O(d^4 m_u m_v)$. Néanmoins nous ne l'avons pas expérimenté, car la structure de données s'avère complexe pour un gain en temps incertain, qui n'est au plus que d'une petite constante multiplicative.

Nous savons maintenant calculer le volume d'un patch exprimé dans V par ses coefficients $\mathbf{p_i}$ (voir (3.3)) : pré-calculer et stocker les coefficients θ^u_{ijk} et θ^v_{ijk} non nuls, puis calculer le volume par la formule (3.6) en utilisant l'expression (3.7) des $\theta_{\mathbf{ijk}}$. Pour appliquer une déformation à volume constant sur un patch il nous faut maintenant calculer le volume de la surface déformée, sachant que la déformation est définie dans l'espace d'approximation $\widetilde{V}$.

Volume de la surface déformée.

Soit $\Delta \in \widetilde{V}$ une déformation (3.4) définie par $\left(\widetilde{\delta}_{\mathbf{i}}\right)_{\mathbf{i}} = \left(\widetilde{\delta}X, \widetilde{\delta}Y, \widetilde{\delta}Z\right) \in (\mathbb{R}^{\widetilde{m}_u \widetilde{m}_v})^3$, c'est-à-dire

$$\widetilde{\delta}X = \left(\widetilde{\delta}x_{\mathbf{i}}\right)_{\mathbf{i}} , \widetilde{\delta}Y = \left(\widetilde{\delta}y_{\mathbf{i}}\right)_{\mathbf{i}} \text{ et } \widetilde{\delta}Z = \left(\widetilde{\delta}z_{\mathbf{i}}\right)_{\mathbf{i}} .$$

Nous avons à calculer le volume de $S^q + \Delta$, sachant que S^q est définie en base fine, alors que Δ est définie en base grossière. En anticipant un peu sur les sections décrivant les déformations contraintes (3.2.3 et 3.2.4), disons que nous allons linéariser la formule du volume en appliquant des déformations axe par axe, comme pour les courbes (section 2.5). Par conséquent il nous suffit d'étudier les cas où Δ est non nulle sur un seule axe. Supposons que la déformation se fasse en X uniquement, *i.e.* $\widetilde{\delta}Y = 0$ et $\widetilde{\delta}Z = 0$. Si l'on reporte l'expression

$$S^q(u,v) + \Delta(u,v) = \sum_{\mathbf{i}=(0,0)}^{(m_u-1, m_v-1)} \mathbf{p_i}\ N_{\mathbf{i}}(u,v) + \sum_{\mathbf{i}=(0,0)}^{(\widetilde{m}_u-1, \widetilde{m}_v-1)} \widetilde{\delta}_{\mathbf{i}} \widetilde{N}_{\mathbf{i}}(u,v)$$

dans la formule (3.5), on obtient le volume de la surface déformée en fonction du volume de la surface initiale et d'un produit scalaire isolant la variable $\widetilde{\delta}X$:

$$\Theta(S^q + \Delta) = \Theta(S^q) + \sum_{\mathbf{i}=(0,0)}^{(\widetilde{m}_u-1, \widetilde{m}_v-1)} \widetilde{\delta}x_{\mathbf{i}}\ \widetilde{\Theta}_{YZ}(\mathbf{i}) = \Theta(S^q) + \langle\ \widetilde{\Theta}_{YZ}\ ,\ \widetilde{\delta}X\ \rangle \qquad (3.12)$$

où

$$\widetilde{\Theta}_{YZ} = \big(\widetilde{\Theta}_{YZ}(\mathbf{i})\big)_{\mathbf{i}} \in \mathbb{R}^{\widetilde{m}_u \widetilde{m}_v}$$

est le vecteur des sommes

$$\widetilde{\Theta}_{YZ}(\mathbf{i}) = \sum_{\mathbf{j},\mathbf{k}} y_{\mathbf{j}}\, z_{\mathbf{k}} \widetilde{\theta}_{\mathbf{ijk}} \quad \text{pour} \quad (0,0) \leq \mathbf{i} \leq (\widetilde{m}_u - 1, \widetilde{m}_v - 1) \tag{3.13}$$

où

$$\widetilde{\theta}_{\mathbf{ijk}} = \int_0^1 N_{j_u} N_{k_u} \widetilde{N}'_{i_u} du \int_0^1 \widetilde{N}_{i_v} N_{k_v} N'_{j_v} dv - \int_0^1 \widetilde{N}_{i_u} N_{k_u} N'_{j_u} du \int_0^1 N_{j_v} N_{k_v} \widetilde{N}'_{i_v} dv \tag{3.14}$$

sont définis comme les $\theta_{\mathbf{ijk}}$ (voir équation (3.7)), mais en mélangeant les fonctions de base de $\widetilde{V}$ et de V.

La difficulté pour mettre en œuvre cette formulation est que les B-splines sous les intégrales sont définies sur des séquences de nœuds différentes : $\mathbf{u}$ ou $\mathbf{v}$ pour les unes, mais $\widetilde{\mathbf{u}}$ ou $\widetilde{\mathbf{v}}$ pour les autres. Par conséquent la méthode récursive présentée pour calculer les $\theta_{\mathbf{ijk}}$ n'est pas applicable telle quelle. Plutôt que de développer une méthode spécifique, nous allons exploiter les formules d'insertion de nœuds pour calculer les $\widetilde{\theta}_{\mathbf{ijk}}$ *à partir* des $\theta_{\mathbf{ijk}}$, puis $\widetilde{\Theta}_{YZ}$ à partir de Θ_{YZ}.

La récurrence de Boehm [Boe80, Far01], rappelée en annexe A.1, permet d'exprimer des relations entre B-splines lorsqu'un nœud est *inséré*. Nous allons légèrement modifier sa formulation pour prendre en compte une *suppression* de nœud. Supposons par exemple que nous supprimions u_l, *i.e.* $\mathbf{u} = \widetilde{\mathbf{u}} \cup \{u_l\}$. Alors les B-splines $\widetilde{N}_k^d$ sur la sous-séquence $\widetilde{\mathbf{u}}$ s'expriment en fonctions des B-splines N_k^d sur la séquence $\mathbf{u}$ par :

$$\widetilde{N}_k^d(u) = \frac{u_{l+1} - u_k}{u_{k+d+1} - u_k} N_k^d(u) + \frac{u_{k+d+2} - u_{l+1}}{u_{k+d+2} - u_{k+1}} N_{k+1}^d(u) \quad \text{pour} \quad l - d \leq k \leq l$$

et $\widetilde{N}_k^d = N_k^d$ sinon.

Nous allons maintenant injecter cette équation dans les formules intégrales (3.14) pour successivement :

- faire remonter (par linéarité de l'intégrale) la récurrence sur les θ^u et θ^v,
- faire remonter (par la formule (3.14)) la récurrence sur les $\theta_{\mathbf{ijk}}$,
- faire remonter (par la formule (3.7)) la récurrence sur Θ_{YZ},
- écrire un algorithme de calcul de $\widetilde{\Theta}_{YZ}$ à partir de Θ_{YZ}.

Supposons, pour cas d'école, que $\widetilde{\mathbf{v}} = \mathbf{v}$. La formule (3.14) devient :

$$\begin{aligned}
\widetilde{\theta}_{\mathbf{ijk}} &= \int N_{j_u} N_{k_u} \widetilde{N}'_{i_u} du \int N_{i_v} N_{k_v} N'_{j_v} dv - \int \widetilde{N}_{i_u} N_{k_u} N'_{j_u} du \int N_{j_v} N_{k_v} N'_{i_v} dv \\
&= \Big(\frac{u_{l+1} - u_{i_u}}{u_{i_u+d+1} - u_{i_u}} \theta^u_{j_u,k_u,i_u} + \frac{u_{i_u+d+2} - u_{l+1}}{u_{i_u+d+2} - u_{i_u}} \theta^u_{j_u,k_u,(i_u+1)} \Big) \theta^v_{i_v k_v j_v} \\
&\quad - \Big(\frac{u_{l+1} - u_{i_u}}{u_{i_u+d+1} - u_{i_u}} \theta^u_{i_u,k_u,j_u} + \frac{u_{i_u+d+2} - u_{l+1}}{u_{i_u+d+2} - u_{i_u}} \theta^u_{i_u+1,k_u,j_u} \Big) \theta^v_{j_v k_v i_v} \\
&= \frac{u_{l+1} - u_{i_u}}{u_{i_u+d+1} - u_{i_u}} \theta_{i_u,i_v,j_u,j_v,k_u,k_v} + \frac{u_{i_u+d+2} - u_{l+1}}{u_{i_u+d+2} - u_{i_u}} \theta_{i_u+1,i_v,j_u,j_v,k_u,k_v} \ .
\end{aligned}$$

Par conséquent, en reportant dans (3.7), pour $l - d \leq i_u \leq l$:

$$\widetilde{\Theta}_{YZ}(i_u, i_v) = \frac{u_{l+1} - u_{i_u}}{u_{i_u+d+1} - u_{i_u}} \Theta_{YZ}(i_u, i_v) + \frac{u_{i_u+d+2} - u_{l+1}}{u_{i_u+d+2} - u_{i_u+1}} \Theta_{YZ}(i_u + 1, i_v) \ . \tag{3.15}$$

Une équation symétrique est obtenue en inversant les rôles de u et v, c'est-à-dire en supprimant un nœud $v_l \in \mathbf{v}$:

$$\widetilde{\Theta}_{YZ}(i_u, i_v) = \frac{v_{l+1} - v_{i_v}}{v_{i_v+d+1} - v_{i_v}} \Theta_{YZ}(i_u, i_v) + \frac{v_{i_v+d+2} - v_{l+1}}{v_{i_v+d+2} - v_{i_v+1}} \Theta_{YZ}(i_u, i_v + 1) \ . \tag{3.16}$$

Remarque 1 : Ces équations présupposent une déformation Δ en X seulement. Les rôles de x, y et z peuvent être permutés, c'est-à-dire en prenant une déformation Δ en Y ou Z, ce qui modifie aussi la formule (3.12) :

$$\Theta(S^q + \Delta) = \Theta(S^q) + \langle\ \widetilde{\Theta}_{XZ}\ ,\ \widetilde{\delta} Y\ \rangle \quad \text{pour} \quad \widetilde{\delta} X = \widetilde{\delta} Z = \vec{0}, \tag{3.17}$$

$$\Theta(S^q + \Delta) = \Theta(S^q) + \langle\ \widetilde{\Theta}_{XY}\ ,\ \widetilde{\delta} Z\ \rangle \quad \text{pour} \quad \widetilde{\delta} X = \widetilde{\delta} Y = 0. \tag{3.18}$$

Remarque 2 : Ces équations représentent la suppression d'un unique nœud : il faudra itérer ce calcul pour tous les nœuds de $\mathbf{u} \setminus \widetilde{\mathbf{u}}$ et $\mathbf{v} \setminus \widetilde{\mathbf{v}}$.

Algorithme de suppression de nœuds.

L'intérêt des formules (3.15) et (3.16) (et de toutes leurs symétriques en Y et Z) est qu'elles nous permettent de construire un algorithme de *suppression de nœuds* pour calculer $\widetilde{\Theta}_{XY}$, $\widetilde{\Theta}_{XZ}$ et $\widetilde{\Theta}_{YZ}$ à partir de Θ_{XY}, Θ_{XZ} et Θ_{YZ}, ce qui nous sera nécessaire pour déformer la surface. Un algorithme pour supprimer tous les nœuds de $\mathbf{u} \setminus \widetilde{\mathbf{u}}$ sans allocation mémoire auxiliaire est proposé ALGO. 3.1. C'est un algorithme unidimensionnel (*i.e.* sur une seule séquence de nœuds) qui doit être appliqué en produit tensoriel, c'est-à-dire :

- pour chaque colonne de $\widetilde{\Theta}_{YZ}$, supprimer chaque nœud de $\mathbf{u} \setminus \widetilde{\mathbf{u}}$,
- puis pour chaque ligne de $\widetilde{\Theta}_{YZ}$, supprimer chaque nœud de $\mathbf{v} \setminus \widetilde{\mathbf{v}}$.

Résumé du calcul de volume pour un patch déformé.

Nous avons maintenant à notre disposition un algorithme de calcul du volume pour un patch $S^q \in V$ déformé par $\Delta \in \widetilde{V}$ selon un certain axe, X par exemple. Cet algorithme comporte les étapes suivantes :

- Pré-calculer et stocker les coefficients θ^u_{ijk} et θ^v_{ijk} non nuls.
- Calculer Θ_{YZ} par la formule (3.10) en utilisant l'expression (3.7) des $\theta_{\mathbf{ijk}}$.
- Calculer $\widetilde{\Theta}_{YZ}$ à partir de Θ_{YZ} par l'algorithme 3.1 de suppression de nœuds appliqué en produit tensoriel.
- Calculer le volume de $S^q + \Delta$ grâce aux formules (3.11) et (3.12) :

$$\Theta(S^q + \Delta) = \langle\ \Theta_{XY}\ ,\ Z\ \rangle + \langle\ \widetilde{\Theta}_{YZ}\ ,\ \widetilde{\delta} X\ \rangle \ .$$

Algo. **3.1** Suppression de nœuds sans allocation mémoire.
Le vecteur $\mathbf{u}$ contient la séquence de nœuds.
Le tableau $\Theta(0 \dots n-1)$ contient initialement les valeurs dans la base fine (*i.e.* une colonne de Θ_{YZ} par exemple).
Au début de chaque boucle $(l-m)$ nœuds ont été supprimés et les valeurs courantes se trouvent dans $\Theta(0, \dots, m-1, l, \dots, n-1)$.
À la sortie $\Theta(0 \dots m-1)$ contient les valeurs dans la base grossière (*i.e.* une colonne de $\widetilde{\Theta}_{YZ}$ par exemple) et $\mathbf{u}(0 \dots m-1)$ contient $\widetilde{\mathbf{u}}$.

$m \leftarrow d+1$ *// les nœuds de bord sont conservés*
pour $l = d+1$ à $n-1$ **faire**
  si $\mathbf{u}(l)$ est conservé **alors**
    $\Theta(m) \leftarrow \Theta(l)$
    $\mathbf{u}(m) \leftarrow \mathbf{u}(l)$
    $m \leftarrow m+1$
  sinon *//* $\mathbf{u}(l)$ *est supprimé*
    pour $r = 0$ à $d-1$ **faire** *//* $m-d-1 \leq m-d-1+r \leq m-2$

$$\Theta(m-d-1+r) \leftarrow \frac{\mathbf{u}(l) - \mathbf{u}(m-d-1+r)}{\mathbf{u}(l+r) - \mathbf{u}(m-d-1+r)}\Theta(m-d-1+r) + \frac{\mathbf{u}(l+r+1) - \mathbf{u}(l)}{\mathbf{u}(l+r+1) - \mathbf{u}(m-d+r)}\Theta(m-d+r)$$

    fin pour

    // cas $r = d$ *traité à part :* $m-d+r = m$ *est remplacé par* l

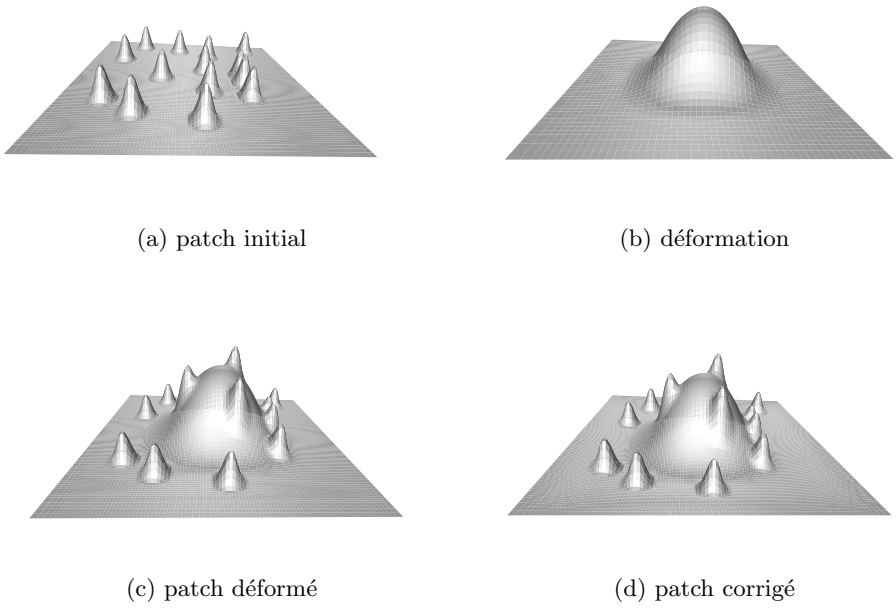
$$\Theta(m-1) \leftarrow \frac{\mathbf{u}(l) - \mathbf{u}(m-1)}{\mathbf{u}(l+d) - \mathbf{u}(m-1)}\Theta(m-1) + \Theta(l)$$

  fin si
fin pour

3.2.3 Correction du volume avec relaxation de la position

Nous venons de construire une méthode de calcul du volume pour un patch B-spline produit tensoriel de $S^q \in V$, ainsi que pour le patch déformé dans $\widetilde{V}$. Nous présentons maintenant une première méthode de déformation sous contrainte de volume. Cette méthode est une *correction* du volume : le patch est d'abord édité *sans* contrainte par un déplacement d'un point du patch, et dans un deuxième temps il est corrigé pour satisfaire la contrainte de volume.

Nous allons d'abord resituer cette étape dans un processus d'édition pour bien définir nos objectifs. Nous proposons ensuite une solution basée sur une minimisation qui peut se résoudre explicitement. Cela nous permet de construire un algorithme de déformation à volume constant. Enfin nous étudions deux améliorations de cet algorithme qui permettent de contrôler l'aspect de la déformation : la répartition du volume selon les axes, et la localisation de la déformation.



(a) patch initial (b) déformation

(c) patch déformé (d) patch corrigé

FIG. 3.4 – Correction du volume.
Le patch initial S^q (a) est édité par ajout d'une large déformation Δ_{pos} (b). La surface $\bar{S}^q$ obtenue (c) est ensuite corrigée pour contraindre le volume (d).

Replaçons nous dans un processus d'édition illustré en FIG. 3.4. Soit S^q un patch de volume $\Theta(S^q) = \Theta_{ref}$. L'utilisateur, qui souhaite le déformer tout en conservant son volume, définit :

- Deux sous-séquences de nœuds $\widetilde{\mathbf{u}}$ et $\widetilde{\mathbf{v}}$ pour contrôler l'étendue de la déformation.
- Un point sur la surface de paramètre $(\bar{u}, \bar{v})$ et un déplacement $\vec{d}$ pour ce point. Cela peut se faire typiquement par une opération de drag-and-drop dans un éditeur. Dans ce cas le paramètre $(\bar{u}, \bar{v})$ est trouvé par projection sur la surface du curseur de la souris, et le déplacement $\vec{d}$ est défini par le mouvement du curseur.

Dans un premier temps nous déformons le patch sans contrainte pour satisfaire le déplacement d'après la méthode expliquée section 3.2.1. La surface résultante

$$\bar{S}^q(u, v) = S^q(u, v) + \Delta_{pos}(u, v)$$

est définie par $(\bar{X}, \bar{Y}, \bar{Z})$ et a pour volume $\Theta(\bar{S}^q) = \bar{\Theta}$ qui est différent de Θ_{ref} a priori.

L'objet de cette section est de corriger le volume, c'est-à-dire de définir une surface (X, Y, Z) qui :

i. est "proche" de $(\bar{X}, \bar{Y}, \bar{Z})$ pour satisfaire autant que possible le déplacement défini par l'utilisateur,

ii. a pour volume Θ_{ref}.

Pour cela nous allons définir une déformation Δ par une minimisation sous contrainte : la fonction objectif est la norme quadratique de la déformation pour satisfaire (i) et la contrainte doit satisfaire (ii). Pour que la contrainte de volume soit linéaire au regard de la minimisation nous allons séparer la déformation Δ définie par $(\widetilde{\delta X}, \widetilde{\delta Y}, \widetilde{\delta Z})$ en trois déformations successives, une par axe :

- Δ_X définie par $(\widetilde{\delta X}, 0, 0)$,
- Δ_Y définie par $(0, \widetilde{\delta Y}, 0)$, et
- Δ_Z définie par $(0, 0, \widetilde{\delta Z})$.

Définition de la déformation selon un axe.

Étudions Δ_X, puis Δ_Y et Δ_Z se déduiront par symétrie. Nous avons donc à définir $(X, Y, Z) = (\bar{X} + \delta X, \bar{Y}, \bar{Z})$, c'est-à-dire trouver $\widetilde{\delta X}$ duquel on déduira δX par insertion de nœuds (voir section 3.2.1). Comme annoncé la fonction objectif est $\|\widetilde{\delta X}\|^2$. Pour écrire la contrainte de volume nous allons reprendre l'expression (3.12) :

$$\Theta(S^q + \Delta_X) = \Theta_{ref} \quad \Longleftrightarrow \quad \langle\, \widetilde{\Theta}_{\bar{Y}\bar{Z}} \,,\, \widetilde{\delta X} \,\rangle = \Theta_{ref} - \bar{\Theta} \,.$$

Le problème à résoudre devient :

$$\min \ \|\widetilde{\delta X}\|^2 \quad \text{sous contrainte} \quad \langle\, \widetilde{\Theta}_{\bar{Y}\bar{Z}} \,,\, \widetilde{\delta X} \,\rangle = \Theta_{ref} - \bar{\Theta} \,. \tag{3.19}$$

En utilisant une fois de plus un multiplicateur de Lagrange λ, on obtient un problème minimax :

$$\max_{\lambda} \min_{\widetilde{\delta X}} \ g\big(\widetilde{\delta X}, \lambda\big) \tag{3.20}$$

avec le lagrangien

$$g\big(\widetilde{\delta X}, \lambda\big) = \|\widetilde{\delta X}\|^2 + \lambda\big(\langle\, \widetilde{\Theta}_{\bar{Y}\bar{Z}} \,,\, \widetilde{\delta X} \,\rangle - (\Theta_{ref} - \bar{\Theta})\big) \,. \tag{3.21}$$

La solution vérifie

$$\vec{\bigtriangledown} g \,=\, \vec{0} \quad \Leftrightarrow \quad 2\widetilde{\delta X} + \lambda \widetilde{\Theta}_{\bar{Y}\bar{Z}} = \vec{0} \quad \text{et} \quad \langle\, \widetilde{\Theta}_{\bar{Y}\bar{Z}} \,,\, \widetilde{\delta X} \,\rangle \,=\, \Theta_{ref} - \bar{\Theta}.$$

Ce système peut se résoudre explicitement par

$$\lambda = \frac{-2(\Theta_{ref} - \bar{\Theta})}{\|\widetilde{\Theta}_{\bar{Y}\bar{Z}}\|^2}$$

et

$$\widetilde{\delta X} = \frac{\Theta_{ref} - \bar{\Theta}}{\|\widetilde{\Theta}_{\bar{Y}\bar{Z}}\|^2} \, \widetilde{\Theta}_{\bar{Y}\bar{Z}} \,. \tag{3.22}$$

Ces formules ont leurs symétriques en Y et Z :

$$\widetilde{\delta}Y = \frac{\Theta_{ref} - \bar{\Theta}}{\|\widetilde{\Theta}_{\bar{X}\bar{Z}}\|^2}\,\widetilde{\Theta}_{\bar{X}\bar{Z}} \quad \text{et} \quad \widetilde{\delta}Z = \frac{\Theta_{ref} - \bar{\Theta}}{\|\widetilde{\Theta}_{\bar{X}\bar{Y}}\|^2}\,\widetilde{\Theta}_{\bar{X}\bar{Y}}\ .$$

En pratique nous allons travailler successivement sur chacun des 3 axes, or ces trois formules compensent *chacune* un écart de volume égal à $\Theta_{ref} - \bar{\Theta}$. Cet écart de volume doit donc être réparti sur les 3 axes, c'est-à-dire que l'on définit $\delta\Theta_X$, $\delta\Theta_Y$ et $\delta\Theta_Z$ tels que

$$\delta\Theta_X + \delta\Theta_Y + \delta\Theta_Z = \Theta_{ref} - \bar{\Theta}\ .$$

Il y a de multiples solutions pour cette répartition, nous y reviendrons un peu plus loin. Ensuite on effectue trois résolutions successives pour déterminer $\widetilde{\delta}X$, $\widetilde{\delta}Y$ et $\widetilde{\delta}Z$, en mettant à jour X, Y et Z au fur et à mesure, ce qui définit entièrement notre méthode de correction de volume :

$$\min\ \|\widetilde{\delta}X\|^2 \quad \text{sous contrainte} \quad \langle\, \widetilde{\Theta}_{\bar{Y}\bar{Z}}\ ,\ \widetilde{\delta}X\,\rangle = \delta\Theta_X$$

$$X = \bar{X} + \delta X \quad \text{et le volume devient } \bar{\Theta} + \delta\Theta_X,$$

$$\min\ \|\widetilde{\delta}Y\|^2 \quad \text{sous contrainte} \quad \langle\, \widetilde{\Theta}_{X\bar{Z}}\ ,\ \widetilde{\delta}Y\,\rangle = \delta\Theta_Y$$

$$Y = \bar{Y} + \delta Y \quad \text{et le volume devient } \bar{\Theta} + \delta\Theta_X + \delta\Theta_Y,$$

$$\min\ \|\widetilde{\delta}Z\|^2 \quad \text{sous contrainte} \quad \langle\, \widetilde{\Theta}_{XY}\ ,\ \widetilde{\delta}Z\,\rangle = \delta\Theta_Z$$

$$Z = \bar{Z} + \delta Z \quad \text{et le volume final est } \bar{\Theta} + \delta\Theta_X + \delta\Theta_Y + \delta\Theta_Z\ =\ \Theta_{ref}.$$

L'algorithme complet de déformation est présenté dans ALGO. 3.2 page 49. Il prend en entrée la position $(\bar{u}, \bar{v})$ du point à déplacer et le déplacement $\vec{d}$. Il consiste donc à calculer une déformation pour le déplacement, puis à effectuer les minimisations que nous venons de décrire.

Aspects algorithmiques.

Une étude détaillée des coûts de chaque opération (colonne de droite ALGO. 3.2) montre que le coût global est $O(m_u m_v)$. Il est donc asymptotiquement optimal. On remarque que toutes les opérations principales sont équivalentes, linéaires par rapport au nombre de points de contrôle.

Répartition de la correction de volume.

Cet algorithme autorise une répartition quelconque de $\Theta_{ref} - \bar{\Theta}$ entre $\delta\Theta_X$, $\delta\Theta_Y$ et $\delta\Theta_Z$. Nous faisons ici trois propositions qui ont des effets différents, en utilisant les composantes du vecteur déplacement $\vec{d} = (d_X, d_Y, d_Z)$:

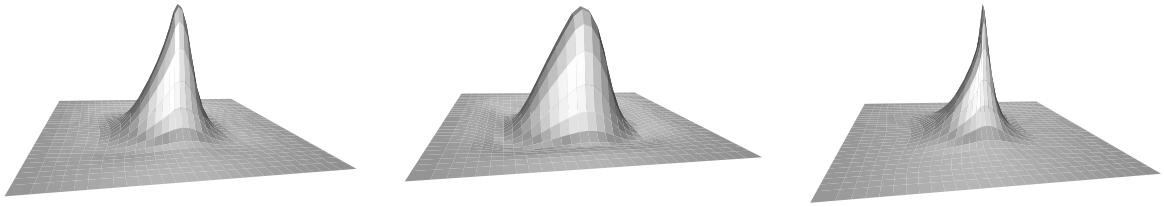
- La répartition équitable, la plus évidente : quel que soit le déplacement les trois axes jouent le même rôle.
$$\delta\Theta_X = \delta\Theta_Y = \delta\Theta_Z = \frac{\Theta_{ref} - \bar{\Theta}}{3}\ .$$
- Une répartition sur chaque axe proportionnellement à la même composantes dans le déplacement $\vec{d}$: par exemple un déplacement en X uniquement sera compensé par une correction en X uniquement, donc dans la même direction. On définit ainsi
$$\delta\Theta_X = \frac{|d_X|^2}{\|\vec{d}\|^2}(\Theta_{ref} - \bar{\Theta})$$
puis $\delta\Theta_Y$ et $\delta\Theta_Z$ en remplaçant d_X par d_Y et d_Z.

Algo. **3.2** Déformation à volume constant.
Une première déformation met la position à jour, puis le volume est corrigé axe par axe.
Le coût global est en $O(m_u m_v d^4) = O(m_u m_v)$.
Paramètres : la position $(\bar{u}, \bar{v})$ du point à déplacer et le déplacement $\vec{d}$.

	Coût
// Position	
$(\delta X, \delta Y, \delta Z) \leftarrow \left(\widetilde{N}_{\mathbf{i}}(\bar{u}, \bar{v})\, \vec{d}\right)_{\mathbf{i}}$	$O(\widetilde{m}_u d^2 + \widetilde{m}_v d^2 + \widetilde{m}_u \widetilde{m}_v)$
$(\delta X, \delta Y, \delta Z) \leftarrow$ `insertion_nœuds`$(\widetilde{\delta X}, \widetilde{\delta Y}, \widetilde{\delta Z})$	$O(m_u m_v d)$
$(X, Y, Z) \leftarrow (X, Y, Z) + (\delta X, \delta Y, \delta Z)$	$O(m_u m_v)$
// Équilibre des volumes	
$\bar{\Theta} \leftarrow$ `calculer_volume` (X, Y, Z)	$O(m_u m_v d^4)$
`calculer` $\delta\Theta_X$, $\delta\Theta_Y$, $\delta\Theta_Z$ *// à partir de* Θ_{ref}, $\bar{\Theta}$ *et* $\vec{d}$	$O(1)$
// Correction du volume par un déplacement en X	
`calculer` Θ_{YZ}	$O(m_u m_v d^2)$
$\widetilde{\Theta}_{YZ} \leftarrow$ `suppression_nœuds`(Θ_{YZ})	$O(m_u m_v d^2)$
$\widetilde{\delta X} \leftarrow \frac{\delta\Theta_X}{\|\widetilde{\Theta}_{YZ}\|^2}\, \widetilde{\Theta}_{YZ}$	$O(\widetilde{m}_u \widetilde{m}_v)$
$\delta X \leftarrow$ `insertion_nœuds`$(\widetilde{\delta X})$	$O(m_u m_v d)$
$X \leftarrow X + \delta X$	$O(m_u m_v)$
// Correction du volume par un déplacement en Y	*idem*
`calculer` Θ_{XZ}	
$\widetilde{\Theta}_{XZ} \leftarrow$ `suppression_nœuds`(Θ_{XZ})	
$\widetilde{\delta Y} \leftarrow \frac{\delta\Theta_Y}{\|\widetilde{\Theta}_{XZ}\|^2}\, \widetilde{\Theta}_{XZ}$	
$\delta Y \leftarrow$ `insertion_nœuds`$(\widetilde{\delta Y})$	
$Y \leftarrow Y + \delta Y$	
// Correction du volume par un déplacement en Z	*idem*
`calculer` Θ_{XY}	
$\widetilde{\Theta}_{XY} \leftarrow$ `suppression_nœuds`(Θ_{XY})	
$\widetilde{\delta Z} \leftarrow \frac{\delta\Theta_Z}{\|\widetilde{\Theta}_{XY}\|^2}\, \widetilde{\Theta}_{XY}$	
$\delta Z \leftarrow$ `insertion_nœuds`$(\widetilde{\delta Z})$	
$Z \leftarrow Z + \delta Z$	

– Une répartition sur chaque axe proportionnellement au déplacement orthogonal : un déplacement en X uniquement sera compensé par une correction en Y et en Z, donc orthogonalement au déplacement. On définit ainsi

$$\delta\Theta_X = \frac{|d_Y|^2 + |d_Z|^2}{2\|\vec{d}\|^2}(\Theta_{ref} - \bar{\Theta}) \quad \text{puis } \delta\Theta_Y \text{ et } \delta\Theta_Z \textit{ mutatis mutandis.}$$



(a) Équitable (b) Proportionnel (c) Orthogonal

FIG. 3.5 – Répartition du volume.
La même déformation est appliquée, le volume est corrigé avec différentes répartitions de $\Theta_{ref} - \bar{\Theta}$: équitable (a), proportionnel au déplacement (b) et proportionnel au déplacement orthogonal (c).

Nous revenons sur cette dernière répartition dans la section 3.2.8 sur les surfaces composites, mais illustrons la brièvement avec la figure 3.5. Un déplacement commun est appliqué pour les trois répartitions. La répartition proportionnelle (b) crée un creux autour du pic, alors que la répartition "orthogonale" (c) va au contraire affiner le pic pour compenser la variation de volume. La répartition équitable (a) est logiquement un mélange des deux.

Déformation locale.

Jusqu'ici nous n'avons raisonné sur une minimisation globale, donc la surface est corrigée globalement, bien que la déformation de position Δ_{pos} soit locale. Il est évidemment intéressant de ne déformer la surface que localement, sur une région qui peut être définie manuellement, ou bien par un paramètre de distance. Notamment, nous avons remarqué que la déformation d'un seul patch à volume constant n'a de sens que si on fixe les courbes de bords, typiquement en vue de l'intégration dans une surface composite fermée. Par conséquent il est important de conserver les points de bords, puisqu'ils définissent à eux seuls les courbes de bord (cf. section 3.2.1).

Pour localiser la déformation, nous proposons d'effectuer la minimisation sur une partie des coefficients seulement. Supposons que les coefficients $\widetilde{\delta}_i$ soient répartis en deux catégories : les coefficients *libres* qui appartiennent à la région du patch que l'on accepte de déformer, et les coefficients *fixes* qui appartiennent au reste de la surface qui ne doit pas bouger. Notons $\mathcal{L} \subset \{\mathbf{i},\ (0,0) \leq \mathbf{i} \leq (\widetilde{m}_u - 1, \widetilde{m}_v - 1)\}$ l'ensemble des indices des coefficients libres. Le raisonnement est identique, mais les dérivées partielles du lagrangien (3.21) sont annulées sur les coefficients libres seulement :

$$\frac{\partial g}{\partial \lambda}(\widetilde{\delta}X, \lambda) = 0 \quad \text{et} \quad \frac{\partial g}{\partial \widetilde{\delta}x_{\mathbf{i}}}(\widetilde{\delta}X, \lambda) = 0 \quad \text{pour tout coefficient } \mathbf{i} \in \mathcal{L}\ .$$

Par conséquent la solution (3.22) et ses symétriques sont modifiés en ce sens que $\widetilde{\Theta}_{\bar{Y}\bar{Z}}(\mathbf{j})$ n'est pris en compte que pour les indices $\mathbf{j}$ des coefficients variables :

$$\widetilde{\delta}x_{\mathbf{i}} = \delta\Theta_X \big(\sum_{\mathbf{j}\in\mathcal{L}} \|\widetilde{\Theta}_{\bar{Y}\bar{Z}}(\mathbf{j})\|^2 \big)^{-1} \widetilde{\Theta}_{\bar{Y}\bar{Z}}(\mathbf{i}) \quad \text{pour tout coefficient } \mathbf{i} \in \mathcal{L}. \tag{3.23}$$

Résumé du processus de déformation d'un patch.

Nous avons maintenant à notre disposition un processus complet de déformation de surfaces composites B-splines non uniformes avec localisation de la déformation.
En entrée :

i. Les sous-séquences de nœuds $\widetilde{\mathbf{u}}$ et $\widetilde{\mathbf{v}}$.

ii. Le déplacement $\vec{d}$ pour le point de paramètre $(\bar{u}, \bar{v})$.

iii. Une répartition de l'écart de volume entre $\delta\Theta_X$, $\delta\Theta_Y$ et $\delta\Theta_Z$.

iv. Une étendue pour la correction de volume est définie, c'est-à-dire un ensemble d'indices $\mathbf{i}$ définissant les coefficients variables.

Le calcul est effectué :

i. Le patch initial (de volume Θ_{ref}) est édité de façon à ce que le déplacement $\vec{d}$ de $S^q(\bar{u}, \bar{v})$ soit effectué.

ii. La correction $\widetilde{\delta}X$ est calculée par (3.23).

iii. La correction $\widetilde{\delta}Y$ est calculée similairement.

iv. La correction $\widetilde{\delta}Z$ est calculée similairement.

Critique.

La méthode que nous venons de présenter a un principal inconvénient. Nous appliquons une première déformation pour la position *puis* les corrections pour le volume. Par conséquent le déplacement $\vec{d}$ défini par l'utilisateur n'est plus vérifié exactement par la surface. On imagine tout de suite la conséquence dans une opération de drag-and-drop : la position de la courbe finale "prend du retard" sur le curseur de la souris.

Une première solution pour corriger cela serait d'ajouter une contrainte linéaire dans les minimisations. Cela complexifie un peu le système linéaire. De façon générale, il est possible d'ajouter n'importe quelle contrainte linéaire, de les intégrer par un multiplicateur de Lagrange, et de résoudre un système linéaire plus gros. Le problème est que nous ne pouvons pas nécessairement trouver une solution explicite comme (3.22), pas de solution générale en tous cas. Ceci nous oblige à résoudre le système par une méthode générale, et nous perdons beaucoup en efficacité.

Une deuxième solution, que nous allons développer dans la prochaine section 3.2.4, est de contraindre en une seule fois la position et le volume. Mais, comme nous le verrons, cela nous ôte la souplesse de répartir librement la correction de volume entre les axes.

3.2.4 Correction du volume avec position stricte

Nous présentons ici une solution alternative à celle de la section 3.2.3, qui traite simultanément la position du point déplacé par l'utilisateur et la contrainte de volume. Nous ne sommes donc plus dans un schéma "déformation puis correction", mais bien dans un schéma "déformation contrainte". Ici aussi nous allons séparer la déformation selon les trois axes. Pour

les mêmes raisons, nous travaillons sur un unique patch, et nous abordons le cas des surfaces composites dans la section 3.2.7.

Partant d'une surface S^q définie par (X, Y, Z), nous cherchons une surface déformée $S^q + \Delta$ définie par $(X + \delta X, Y + \delta Y, Z + \delta Z)$. L'utilisateur définit deux sous-séquences de nœuds $\widetilde{\mathbf{u}}$ et $\widetilde{\mathbf{v}}$ (voir section 3.2.1) qui déterminent l'échelle de la déformation. Il définit ensuite un déplacement $\vec{d}$ pour un point de paramètre $(\bar{u}, \bar{v})$. Il nous faut alors calculer la déformation $(\widetilde{\delta X}, \widetilde{\delta Y}, \widetilde{\delta Z})$ dans la base grossière de façon à vérifier :

i. la contrainte de volume $\Theta(S^q + \Delta) = \Theta(S^q)$, et

ii. la contrainte de position $\Delta(\bar{u}, \bar{v}) = \vec{d}$.

En écrivant la déformation dans la base grossière

$$\Delta(u, v) = \sum_{\mathbf{j}} \widetilde{\delta}_{\mathbf{j}} \widetilde{N}_{\mathbf{j}}(u, v),$$

la contrainte de position (ii) se réécrit :

$$\sum_{\mathbf{j}} \widetilde{\delta}_{\mathbf{j}} \widetilde{N}_{\mathbf{j}}(\bar{u}, \bar{v}) = \vec{d}.$$

Autrement dit, pour adopter des notations vectorielles plus pratiques, en posant $\widetilde{N}_{\bar{u}\bar{v}} = \left(\widetilde{N}_{\mathbf{i}}(\bar{u}, \bar{v})\right)_{\mathbf{i}}$, la contrainte s'écrit axe par axe sous forme de produit scalaire :

$$\begin{cases} \langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\delta X} \,\rangle = d_X \\ \langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\delta Y} \,\rangle = d_Y \\ \langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\delta Z} \,\rangle = d_Z \end{cases} \tag{3.24}$$

où d_X, d_Y et d_Z sont les composantes de $\vec{d}$.
Le calcul de $\widetilde{N}_{\bar{u}\bar{v}}$ se fait en deux étapes :

- Calculer $\left(\widetilde{N}_{i_u}(\bar{u})\right)_{i_u}$ et $\left(\widetilde{N}_{i_v}(\bar{v})\right)_{i_v}$ par l'algorithme de De Boor unidimensionnel, ce qui coûte $O(\widetilde{m}_u d^2 + \widetilde{m}_v d^2)$.
- Calculer les $\widetilde{N}_{\mathbf{i}}(\bar{u}, \bar{v}) = \widetilde{N}_{i_u}(\bar{u}) \widetilde{N}_{i_v}(\bar{v})$, ce qui coûte $O(\widetilde{m}_u \widetilde{m}_v)$.

De son côté, la contrainte de volume (i) s'exprime par les produits scalaires :

$$\begin{cases} \langle\, \widetilde{\Theta}_{YZ} \,,\, \widetilde{\delta X} \,\rangle = 0 \\ \langle\, \widetilde{\Theta}_{XZ} \,,\, \widetilde{\delta Y} \,\rangle = 0 \\ \langle\, \widetilde{\Theta}_{XY} \,,\, \widetilde{\delta Z} \,\rangle = 0 \end{cases} \tag{3.25}$$

Nous avons deux contraintes scalaires (3.24) et (3.25) pour chaque axe, et autant de degrés de liberté que de points de contrôle variables. Nous optons de nouveau pour une minimisation de la norme quadratique sous contrainte, ce qui nous amène au processus suivant, au cours duquel le volume est constant :

$$\min \ \|\widetilde{\delta X}\|^2 \quad \text{sous contrainte} \quad \begin{cases} \langle\, \widetilde{\Theta}_{YZ} \,,\, \widetilde{\delta X} \,\rangle = 0 \\ \langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\delta X} \,\rangle = d_X \end{cases}$$

$$X \ \leftarrow \ X + \delta X \quad \text{dès lors la position en } X \text{ est vérifiée,}$$

$$\min \|\widetilde{\delta}Y\|^2 \quad \text{sous contrainte} \quad \begin{cases} \langle\, \widetilde{\Theta}_{XZ} \,,\, \widetilde{\delta}Y \,\rangle = 0 \\ \langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\delta}Y \,\rangle = d_Y \end{cases}$$

$$Y \leftarrow Y + \delta Y \quad \text{dès lors la position en } Y \text{ est vérifiée,}$$

$$\min \|\widetilde{\delta}Z\|^2 \quad \text{sous contrainte} \quad \begin{cases} \langle\, \widetilde{\Theta}_{XY} \,,\, \widetilde{\delta}Z \,\rangle = 0 \\ \langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\delta}Z \,\rangle = d_Z \end{cases}$$

$$Z \leftarrow Z + \delta Z \quad \text{dès lors la position en } Z \text{ est vérifiée.}$$

Étudions la première minimisation, les autres se déduiront par symétrie. En utilisant les multiplicateurs de Lagrange λ et μ, elle se transforme en problème minimax :

$$\max_{\lambda,\mu} \min_{\widetilde{\delta}X} \|\widetilde{\delta}X\|^2 + \lambda\langle\, \widetilde{\Theta}_{YZ} \,,\, \widetilde{\delta}X \,\rangle + \mu(\langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\delta}X \,\rangle - d_X)$$

$$\Leftrightarrow \quad \vec{\nabla}\Big(\|\widetilde{\delta}X\|^2 + \lambda\langle\, \widetilde{\Theta}_{YZ} \,,\, \widetilde{\delta}X \,\rangle + \mu(\langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\delta}X \,\rangle - d_X)\Big) = \vec{0}$$

$$\Leftrightarrow \quad \begin{cases} 2\widetilde{\delta}X + \lambda\widetilde{\Theta}_{YZ} + \mu\widetilde{N}_{\bar{u}\bar{v}} = \vec{0} \\ \langle\, \widetilde{\Theta}_{YZ} \,,\, \widetilde{\delta}X \,\rangle = 0 \\ \langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\delta}X \,\rangle = d_X \end{cases}$$

En supposant que $\widetilde{N}_{\bar{u}\bar{v}}$ et $\widetilde{\Theta}_{YZ}$ ne sont pas colinéaires, le système est de rang plein. La solution s'écrit alors :

$$\lambda = \frac{2\, d_X \,\langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\Theta}_{YZ} \,\rangle}{\|\widetilde{N}_{\bar{u}\bar{v}}\|^2 \,\|\widetilde{\Theta}_{YZ}\|^2 - \langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\Theta}_{YZ} \,\rangle^2} \;,$$

$$\mu = \frac{-2\, d_X \,\|\widetilde{\Theta}_{YZ}\|^2}{\|\widetilde{N}_{\bar{u}\bar{v}}\|^2 \,\|\widetilde{\Theta}_{YZ}\|^2 - \langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\Theta}_{YZ} \,\rangle^2} \;,$$

et

$$\widetilde{\delta}X = \frac{d_X}{\|\widetilde{N}_{\bar{u}\bar{v}}\|^2 \,\|\widetilde{\Theta}_{YZ}\|^2 - \langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\Theta}_{YZ} \,\rangle^2}\Big(\|\widetilde{\Theta}_{YZ}\|^2 \widetilde{N}_{\bar{u}\bar{v}} - \langle\, \widetilde{N}_{\bar{u}\bar{v}} \,,\, \widetilde{\Theta}_{YZ} \,\rangle \widetilde{\Theta}_{YZ}\Big). \tag{3.26}$$

Des formules symétriques sont obtenues pour Y et Z. Nous pouvons maintenant formaliser le processus global par l'algorithme 3.3 page 54. Le coût global de cet algorithme est $O(m_u m_v)$, c'est-à-dire linéaire par rapport au nombre de points de contrôle.

Déformation locale.

Comme nous l'avons fait pour l'algorithme de la section 3.2.3, il est possible de faire des minimisations partielles pour localiser la déformation. Il s'agit de restreindre la minimisation à l'ensemble des points de contrôles que l'on souhaite voir modifiés. Cela se traduit dans la formule résultat (3.26) par une restriction aux coefficients variables pour $\widetilde{\delta}X$ et $\widetilde{\Theta}_{YZ}$, mais aussi pour $\widetilde{N}_{\bar{u}\bar{v}}$.

Commentaires.

Comme nous l'avons déjà souligné, la principale différence entre les algorithmes proposés dans les sections 3.2.3 et 3.2.4 est que le premier offre la possibilité de répartir librement la correction de volume entre les trois axes, alors que le deuxième a l'avantage de satisfaire exactement le déplacement $\vec{d}$. Nous ferons d'avantage de comparaisons avec des illustrations lorsque nous aurons généralisé ces algorithmes aux surfaces composites.

ALGO. **3.3** Déformation à volume constant.
La position et la contrainte de volume sont traités simultanément pour chaque axe.
Le coût global est en $O(m_u m_v d^2) = O(m_u m_v)$.
Paramètres : la position $(\bar{u}, \bar{v})$ du point à déplacer et le déplacement $\vec{d}$.

	Coût
`calculer` $\widetilde{N}_{\bar{u}\bar{v}}$	$O(\widetilde{m}_u d^2 + \widetilde{m}_v d^2 + \widetilde{m}_u \widetilde{m}_v)$
// Déformation en X	
`calculer` Θ_{YZ}	$O(m_u m_v d^2)$
$\widetilde{\Theta}_{YZ} \leftarrow$ `suppression_nœuds`(Θ_{YZ})	$O(m_u m_v d)$
`calculer` $\widetilde{\delta X}$	$O(\widetilde{m}_u \widetilde{m}_v)$
$\delta X \leftarrow$ `insertion_nœuds`$(\widetilde{\delta X})$	$O(m_u m_v d)$
$X \leftarrow X + \delta X$	$O(m_u m_v)$
// Déformation en Y	*idem*
`calculer` Θ_{XZ}	
$\widetilde{\Theta}_{XZ} \leftarrow$ `suppression_nœuds`(Θ_{XZ})	
`calculer` $\widetilde{\delta Y}$	
$\delta Y \leftarrow$ `insertion_nœuds`$(\widetilde{\delta Y})$	
$Y \leftarrow Y + \delta Y$	
// Déformation en Z	*idem*
`calculer` Θ_{XY}	
$\widetilde{\Theta}_{XY} \leftarrow$ `suppression_nœuds`(Θ_{XY})	
`calculer` $\widetilde{\delta Z}$	
$\delta Z \leftarrow$ `insertion_nœuds`$(\widetilde{\delta Z})$	
$Z \leftarrow Z + \delta Z$	

3.2.5 Surfaces composites

Nous venons de construire deux algorithmes de complexité équivalente pour déformer un patch B-spline produit tensoriel en conservant ce que nous avons appelé le "volume du patch". Nous avons souligné que ceci avait un sens tant que les courbes de bord du patch sont fixes, dans la perspective d'une inclusion dans une surface composite.

Dans un premier temps, considérant une telle surface continue et fermée, modifier un seul patch par les précédentes méthodes est une déformation de la surface à volume constant. Dans un deuxième temps, il est intéressant de pouvoir déformer plusieurs patchs de la même surface. Pour cela nous allons généraliser les deux algorithmes, mais il nous faut avant tout aborder deux questions : les difficultés liées aux séquences de nœuds entre deux patchs voisins, et la définition d'un voisinage inter-patchs.

Cohérence des séquences pour l'édition multi-échelle.

Soit S une surface composite faite d'une collection de patchs $\{S^q\}_q$ qui se recollent suivant leurs bords $u \equiv 0$, $u \equiv 1$, $v \equiv 0$ et $v \equiv 1$. Dans la base fine, pour assurer la continuité de la surface sur une *couture* (bord commun entre deux patchs), il ne suffit pas d'assurer l'égalité des points de contrôle de part et d'autre de la couture. Il faut aussi vérifier que les séquences de nœuds correspondent. Précisons ce que cela signifie.

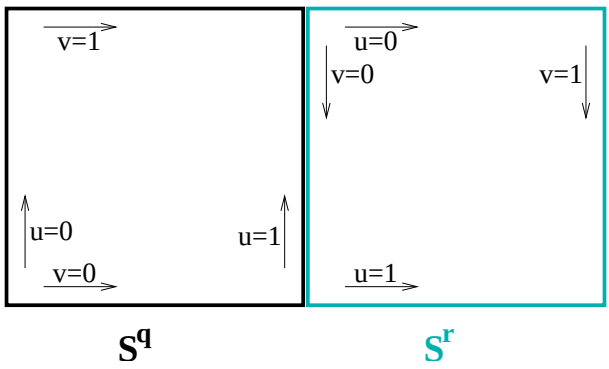


Fig. 3.6 – Paramétrisation le long des coutures.

Le long d'une couture entre deux patchs S^q et S^r, chacun des deux peut présenter n'importe lequel de ses 4 bords, ce qui fait 16 possibilités, 10 si l'on prend en compte la symétrie. Par exemple sur la figure 3.6, le bord $u \equiv 1$ du patch S^q jouxte le bord $v \equiv 0$ du patch S^r. Examinons maintenant le paramètre *variable* le long de la couture : v pour S^q et u pour S^r, mais les deux varient en sens inverse, ce qui impose aux séquences $\mathbf{u}$ et $\mathbf{v}$ d'être symétriques l'une de l'autre. Dans la mesure où les deux patchs ont la même orientation (pour pouvoir définir un intérieur à la surface composite), il y a en fait 6 possibilités :

(a) u face à u qui varient dans le même sens,
(b) u face à u qui varient en sens inverse,
(c) v face à v qui varient dans le même sens,
(d) v face à v qui varient en sens inverse,
(e) u face à v qui varient dans le même sens,
(f) u face à v qui varient en sens inverse.

Pour rester général, il faut envisager chaque éventualité, ce qui conduit aux contraintes suivantes :

i. Les degrés des B-splines en u et en v doivent être les mêmes.

ii. Les cas (b) et (d) imposent que les séquences soient symétriques, c'est-à-dire

$$u_{m_u+d-i_u} = 1 - u_{i_u} \quad \text{et} \quad v_{m_v+d-i_v} = 1 - v_{i_v} \ .$$

iii. Le cas (e) impose $\mathbf{u} = \mathbf{v}$.

La satisfaction de ces conditions est une condition suffisante pour affirmer : si les points de contrôle sur la couture sont égaux (entre les deux patchs), alors le raccord est continu.

Lors de la construction d'une déformation (dans l'espace grossier $\widetilde{V}$) qui traverse une couture, les sous-séquences $\widetilde{\mathbf{u}}$ et $\widetilde{\mathbf{v}}$ devront aussi vérifier les conditions (i), (ii) et (iii). Notons néanmoins que ces conditions ne s'appliquent que dans la région que l'on souhaite déformer, et que, selon l'agencement des patchs, elles ne s'appliquent pas forcément à toutes. Dans la suite nous supposons que toutes ces contraintes de cohérence sont satisfaites.

Sélection du voisinage $\mathcal{L}$ définissant l'étendue de la déformation.

Dans une interface haut niveau, l'utilisateur visualise la surface composite, mais n'a pas nécessairement accès au découpage en patchs. En tout cas, il n'est pas forcément souhaitable qu'il les manipule explicitement. Notre idée d'une interface ergonomique est la suivante : lorsque l'utilisateur sélectionne un point sur la surface, il doit pouvoir sélectionner facilement un voisinage autour du point qu'il édite, c'est-à-dire indépendamment du découpage en patchs. C'est ce voisinage qui définit l'étendue de la déformation, correspondant aux coefficients libres pour la minimisation.

Soit $(\bar{u}, \bar{v})$ les paramètres du point modifié sur le patch S^q. Notons $\bar{\mathbf{p}}$ le point de contrôle (grossier) qui a le plus d'influence sur $S^q(\bar{u}, \bar{v})$. Il faut définir un ensemble de tous les points les plus proches de $\bar{\mathbf{p}}$ sur tout le maillage de contrôle grossier. Cet ensemble est noté $\mathcal{L}$. Un point $\mathbf{p}$ du maillage grossier est dans le voisinage $\mathcal{L}$ si il est à une distance de $\bar{\mathbf{p}}$ inférieure à un certain seuil. Il reste à définir cette mesure de distance. Pour cela il semble judicieux de choisir une distance "sur le maillage" plutôt qu'une distance géométrique dans $\mathbb{R}^3$. Nous définissons un *chemin* entre deux points de contrôle sur le maillage comme une suite d'arêtes incidentes qui relient les deux points.

Soit un plus court chemin entre $\mathbf{p}$ et $\bar{\mathbf{p}}$ composé respectivement de ℓ_u et ℓ_v arêtes dans les directions u et v. Nous avons testé trois critères :

– La norme infinie (l'utilisateur doit spécifier e_u et e_v) :

$$\mathcal{L} = \{\mathbf{p},\ \ell_u < e_u \text{ et } \ell_v < e_v\}.$$

– La norme 1 (l'utilisateur doit spécifier e) :

$$\mathcal{L} = \{\mathbf{p},\ \ell_u + \ell_v < e\}.$$

– La norme 2 (l'utilisateur doit spécifier e) :

$$\mathcal{L} = \{\mathbf{p},\ \ell_u^2 + \ell_v^2 < e^2\}.$$

Ce choix est illustré figure 3.7. La même déformation est utilisée pour un voisinage en norme infinie, 1 et 2 (de gauche à droite). La plus intuitive est la norme 2, car le voisinage est circulaire. Mais la norme la plus appropriée à notre modèle est la norme infinie, car les B-splines produit

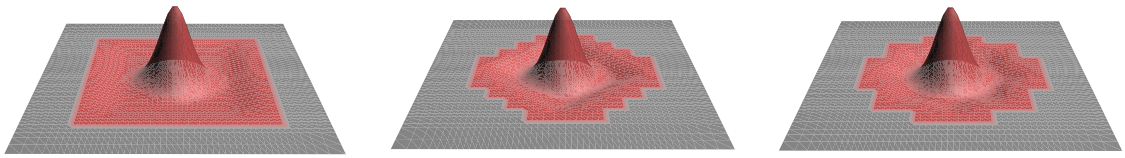


Fig. 3.7 – Comparaison des voisinages.
Les régions déformées sont colorées. De gauche à droite : la norme infinie, la norme 1 et la norme 2.

tensoriel ne sont pas isotropes, donc les normes 1 et infinie donnent des surfaces résultantes qui oscillent légèrement : sur le patch de droite Fig. 3.7, on observe des ombres irrégulières au bord de la région déformée.

Ces calculs de voisinages ne posent aucun problème tant qu'on reste à l'intérieur d'un patch. Dès que l'on traverse une couture il faut faire attention à éventuellement inverser u et v, et/ou inverser les sens de variation des indices. La principale difficulté apparaît au passage d'un angle entre plusieurs patchs. Si le nombre de patchs ayant un angle en commun est pair cela reste cohérent. S'il est impair le problème est plus mal spécifié car un point peut présenter plusieurs plus courts chemins avec ℓ_u et ℓ_v inversés. Dans ce cas nous avons choisi de conserver les deux chemins possibles et de faire l'union des deux étendues engendrées.

D'un point de vue algorithmique, le calcul du voisinage peut se faire par propagation à partir de $\bar{\mathbf{p}}$ à l'aide d'une file de priorité triée par ordre croissant de distance, ce qui permet un calcul en temps linéaire.

Pour augmenter les possibilités d'utilisation, on peut envisager que la région d'influence soit dessinée sur la surface grâce à l'éditeur. On aurait alors toute liberté quant à sa forme.

3.2.6 Extension de la méthode avec relaxation de la position.

Nous présentons ici l'extension de la méthode pour un patch présentée section 3.2.3. Soit S une surface composite faite d'une collection de patchs $\{S^q\}_q$. Au départ du processus d'édition, l'utilisateur définit un point de paramètre $(\bar{u}, \bar{v})$ sur un patch d'indice $\bar{q}$, ainsi qu'un déplacement $\vec{d}$ pour ce point. Ce patch est dans un premier temps déformé selon la formule de Δ_{pos} présentée section 3.2.1. La nouvelle surface composite a un nouveau volume $\Theta_{ref} + \delta V$. Nous supposons connue l'étendue de la déformation, c'est-à-dire un ensemble $\mathcal{L}$ de coefficients dits *libres*, qui peuvent appartenir à des patchs différents.

À partir de là il nous faut corriger le volume, *i.e.* définir la déformation Δ dont le support est réduit aux points libres, telle que le volume final est Θ_{ref}. Pour cela nous adaptons la méthode 3.2.3. Nous allons faire trois minimisations, une par axe. Supposons que l'on traite un de ces axes, quelconque fixé. Notons $\widetilde{\delta}^q \in \mathbb{R}^{\widetilde{m}_u \widetilde{m}_v}$ la déformation associée à cet axe pour chaque patch q, *i.e.* $\widetilde{\delta}^q = \widetilde{\delta} X^q$, $\widetilde{\delta} Y^q$ ou $\widetilde{\delta} Z^q$. Les éléments de la minimisation peuvent être directement étendus de la manière suivante :

i. La fonction objectif est la norme quadratique des déformations sur toute la surface $S = \{S^q\}_q$, *i.e.* la somme des normes quadratiques des déformations sur les patchs

$$\sum_{\text{patchs } q} \|\widetilde{\delta}^q\|^2$$

à laquelle il faut retrancher les points de couture parce qu'ils interviennent plusieurs fois : un point $\mathbf{p}$ sur la couture entre S^q et S^r est comptabilisé dans δ^q *et* dans δ^r.

ii. La contrainte de volume est

$$\delta V = \Theta(S+\Delta) - \Theta(S) = \sum_q \Theta(S^q + \Delta^q) - \Theta(S^q) = \sum_q \langle\, \widetilde{\Theta}^q \,,\, \widetilde{\delta}^q \,\rangle,$$

où $\widetilde{\Theta}^q = \widetilde{\Theta}^q_{YZ}$, $\widetilde{\Theta}^q_{XZ}$ ou $\widetilde{\Theta}^q_{XY}$ selon l'axe sur lequel on travaille (voir équation (3.13) page 43).

iii. Les contraintes de continuité sur les coutures sont des égalités simples entre points de contrôle de part et d'autre de la couture.

On voit tout de suite la difficulté à formuler le problème comme cela. La fonction objectif ne s'écrit pas simplement, et il y a beaucoup de contraintes linéaires pour assurer la continuité sur les coutures. Nous allons donc "retourner" les notations :

- plutôt que de décomposer la déformation en $\widetilde{\delta}^q$ selon les patchs q, et d'ajouter des conditions sur certains points $\mathbf{p}$ sur les coutures,
- nous allons décomposer la déformation en $\widetilde{\delta}_{\mathbf{p}}$ selon les points de contrôle grossiers $\mathbf{p}$, en ajoutant la possibilité pour un point d'intervenir dans plusieurs patchs.

Cette vision des choses revient à formuler le problème ainsi :

i. La fonction objectif est la somme des normes quadratiques sur les sommets libres, quel que soit le nombre de patchs dans lesquels le sommet intervient :

$$\sum_{\mathbf{p}\in\mathcal{L}} |\widetilde{\delta}_{\mathbf{p}}|^2$$

à laquelle il n'y a plus rien à retrancher, car chaque point n'intervient qu'une fois, couture ou pas. $\widetilde{\delta}_{\mathbf{p}} \in \mathbb{R}$ est le déplacement du point de contrôle $\mathbf{p} \in \mathcal{L}$ selon l'axe sur lequel on travaille.

ii. La contrainte de volume est

$$\delta V = \sum_{\mathbf{p}\in\mathcal{L}} \sum_{q/\mathbf{p}\in S^q} \widetilde{\Theta}^q(\mathbf{p})\ \widetilde{\delta}_{\mathbf{p}}$$

où la somme intérieure prend en compte tous les patchs q auxquels $\mathbf{p}$ appartient : cette somme contient un seul terme si $\mathbf{p}$ est à l'intérieur d'un patch, deux termes si $\mathbf{p}$ est sur une couture, et plus si $\mathbf{p}$ est à un angle entre plusieurs patchs.

iii. Les contraintes de continuité "disparaissent" dans cette formulation : si un point $\mathbf{p}$ appartient à deux patchs voisins, il n'est pris en compte qu'une fois dans les expressions ci-dessus, mais $\widetilde{\delta}_{\mathbf{p}}$ servira à mettre à jour les deux patchs (plusieurs si c'est un angle).

Vu ainsi la minimisation se formule :

$$\min \sum_{\mathbf{p}\in\mathcal{L}} |\widetilde{\delta}_{\mathbf{p}}|^2 \quad \text{sous contrainte} \quad \sum_{\mathbf{p}\in\mathcal{L}} \Big(\sum_{q/\mathbf{p}\in S^q} \widetilde{\Theta}^q(\mathbf{p}) \Big) \widetilde{\delta}_{\mathbf{p}} = \delta V \ .$$

En notant $\widetilde{\Theta}(\mathbf{p})$ la somme

$$\widetilde{\Theta}(\mathbf{p}) = \sum_{q/\mathbf{p}\in S^q} \widetilde{\Theta}^q(\mathbf{p}) \ \in \mathbb{R} \ ,$$

elle se reformule avec un multiplicateur de Lagrange λ :

$$\vec{\bigtriangledown} \Big(\sum_{\mathbf{p}\in\mathcal{L}} |\widetilde{\delta}_{\mathbf{p}}|^2 \ + \ \lambda \big(\sum_{\mathbf{p}\in\mathcal{L}} \widetilde{\delta}_{\mathbf{p}} \widetilde{\Theta}(\mathbf{p}) - \delta V \big) \Big) \ = \ \vec{0}$$

équivalent à

$$\begin{cases} 2\widetilde{\delta}_{\mathbf{p}} + \lambda \widetilde{\Theta}(\mathbf{p}) & = 0 \qquad \text{pour tout } \mathbf{p} \in \mathcal{L} \\ \\ \sum \widetilde{\delta}_{\mathbf{p}} \widetilde{\Theta}(\mathbf{p}) & = \delta V \ . \end{cases}$$

La solution de ce système est

$$\lambda = \frac{-2\delta V}{\|\widetilde{\Theta}\|^2} \quad \text{et} \quad \widetilde{\delta}_{\mathbf{p}} = \frac{\delta V}{\|\widetilde{\Theta}\|^2} \widetilde{\Theta}(\mathbf{p}) \quad \text{pour tout } \mathbf{p} \in \mathcal{L}.$$

Comme annoncé, cette solution est utilisée trois fois (généralisant l'algorithme 3.2 aux surfaces composites) :

- avec $\delta V = \delta V_X$ et $\widetilde{\Theta} = \widetilde{\Theta}_{YZ}$ pour calculer la déformation $\widetilde{\delta} X$ selon X,
- avec $\delta V = \delta V_Y$ et $\widetilde{\Theta} = \widetilde{\Theta}_{XZ}$ pour calculer la déformation $\widetilde{\delta} Y$ selon Y,
- avec $\delta V = \delta V_Z$ et $\widetilde{\Theta} = \widetilde{\Theta}_{XY}$ pour calculer la déformation $\widetilde{\delta} Z$ selon Z.

Comme dans la section 3.2.3 il est possible d'équilibrer $\delta V_X + \delta V_Y + \delta V_Z = \delta V$. Nous illustrons et comparons ces résultats en section 3.2.8.

3.2.7 Extension de la méthode avec position stricte.

Nous proposons ici une extension de la méthode avec position stricte pour un patch de la section 3.2.4 aux surfaces composites de la même manière que la section précédente.

Soit S une surface composite faite d'une collection de patchs $\{S^q\}_q$. Pour causer une déformation l'utilisateur définit un point de paramètre $(\bar{u}, \bar{v})$ sur un patch d'indice $\bar{q}$, ainsi qu'un déplacement $\vec{d}$ pour ce point. Nous supposons connue l'étendue de la déformation, c'est-à-dire un ensemble $\mathcal{L}$ de points de contrôles dits *libres* qui peuvent appartenir des patchs différents. Celle-ci peut être définie selon un des critères développés dans la section 3.2.5. Nous allons définir une déformation $\Delta(u, v) \in \widetilde{V}$ de façon à :

- déplacer le point de paramètre $(\bar{u}, \bar{v})$ de $\vec{d} = \Delta(\bar{u}, \bar{v})$, et
- conserver le volume $\Theta(S + \Delta) = \Theta(S)$.

Pour cela nous effectuons une minimisation sous contraintes selon chaque axe. En utilisant les notations de la section 3.2.6, les éléments de ces minimisations sont :

i. La fonction objectif est la somme des normes quadratiques des déplacements :

$$\sum_{\mathbf{p}\in\mathcal{L}} |\widetilde{\delta}_{\mathbf{p}}|^2 \ ,$$

où $\widetilde{\delta}_{\mathbf{p}} \in \mathbb{R}$ est le déplacement du point de contrôle $\mathbf{p} \in \mathcal{L}$ selon l'axe sur lequel on travaille.

ii. La contrainte de volume est

$$0 = \sum_{\mathbf{p}\in\mathcal{L}} \sum_{q/\mathbf{p}\in S^q} \widetilde{\Theta}^q(\mathbf{p})\ \widetilde{\delta}_{\mathbf{p}} \ .$$

où $\widetilde{\Theta}^q = \widetilde{\Theta}^q_{YZ}$, $\widetilde{\Theta}^q_{XZ}$ ou $\widetilde{\Theta}^q_{XY}$ selon l'axe sur lequel on travaille.

iii. La contrainte de position ne concerne que le patch $\bar{q}$ auquel appartient le point sélectionné $S^{\bar{q}}(\bar{u},\bar{v})$. Comme en section 3.2.4 elle s'écrit :

$$d = \sum_{\mathbf{p}\in S^{\bar{q}}} \widetilde{N}^{\bar{q}}_{\mathbf{p}}(\bar{u},\bar{v})\widetilde{\delta}_{\mathbf{p}} \ .$$

Où d est une des trois composantes de $\vec{d}$.

Par conséquent la minimisation s'écrit :

$$\min \sum_{\mathbf{p}\in\mathcal{L}} |\widetilde{\delta}_{\mathbf{p}}|^2 \quad \text{sous les contrainte} \quad \begin{cases} \sum_{\mathbf{p}} \widetilde{N}^{\bar{q}}(\mathbf{p})\widetilde{\delta}_{\mathbf{p}} = d & \text{somme pour } \mathbf{p} \in S^{\bar{q}} \\ \sum_{\mathbf{p}} \sum_{q} \widetilde{\Theta}^q(\mathbf{p})\widetilde{\delta}_{\mathbf{p}} = 0 & \text{somme pout tout } \mathbf{p} \text{ libre} \\ & \text{somme pour } q \ / \ \mathbf{p} \in S^q \end{cases} .$$

En contractant la notation somme

$$\widetilde{\Theta}(\mathbf{p}) = \sum_{q/\mathbf{p}\in S^q} \widetilde{\Theta}^q(\mathbf{p}),$$

et en utilisant les multiplicateurs de Lagrange λ et μ le problème devient

$$\vec{\nabla}\Big(\sum_{\mathbf{p}\in\mathcal{L}} |\widetilde{\delta}_{\mathbf{p}}|^2 \ + \ \lambda \sum_{\mathbf{p}\in\mathcal{L}} \widetilde{\Theta}(\mathbf{p})\widetilde{\delta}_{\mathbf{p}} \ + \ \mu\big(\sum_{\mathbf{p}\in\mathcal{L}} \widetilde{N}^{\bar{q}}(\mathbf{p})\widetilde{\delta}_{\mathbf{p}} - d\big)\Big) \ = \ \vec{0}$$

équivalent à

$$\begin{cases} 2\widetilde{\delta}_{\mathbf{p}} + \lambda\widetilde{\Theta}(\mathbf{p}) + \mu\widetilde{N}^{\bar{q}}(\mathbf{p}) & = 0 \quad \text{pout tout } \mathbf{p} \text{ libre dans } S^{\bar{q}}, \\ 2\widetilde{\delta}_{\mathbf{p}} + \lambda\widetilde{\Theta}(\mathbf{p}) & = 0 \quad \text{pour tout autre } \mathbf{p} \text{ libre}, \\ \sum \widetilde{\Theta}(\mathbf{p})\widetilde{\delta}_{\mathbf{p}} & = 0 \\ \sum \widetilde{N}^{\bar{q}}(\mathbf{p})\widetilde{\delta}_{\mathbf{p}} & = d \ . \end{cases}$$

Alors avec les notations

$$\|\widetilde{\Theta}\|^2 = \sum_{\mathbf{p}\in\mathcal{L}} \big(\widetilde{\Theta}(\mathbf{p})\big)^2, \quad \|\widetilde{N}^{\bar{q}}\|^2 = \sum_{\mathbf{p}\in S^{\bar{q}}} \big(\widetilde{N}^{\bar{q}}(\mathbf{p})\big)^2 \quad \text{et} \quad \sigma_{\bar{q}} = \sum_{\mathbf{p}\in S^{\bar{q}}} \widetilde{\Theta}(\mathbf{p})\widetilde{N}^{\bar{q}}(\mathbf{p}),$$

on obtient la solution en calculant pour les multiplicateurs de Lagrange de la manière suivante :

$$\lambda = \frac{2\,d\,\sigma_{\bar{q}}}{\|\widetilde{N}^{\bar{q}}\|^2\,\|\widetilde{\Theta}\|^2 - \sigma_{\bar{q}}^2} \quad , \qquad \mu = \frac{-2\,d\,\|\widetilde{\Theta}\|^2}{\|\widetilde{N}^{\bar{q}}\|^2\,\|\widetilde{\Theta}\|^2 - \sigma_{\bar{q}}^2} \ .$$

Selon que $\mathbf{p}$ est dans $S^{\bar{q}}$ ou pas, on effectue le déplacement $\delta_{\mathbf{p}}$:

$$\widetilde{\delta}_{\mathbf{p}} = \frac{d}{\|\widetilde{N}^{\bar{q}}\|^2\,\|\widetilde{\Theta}\|^2 - \sigma_{\bar{q}}^2}\big(\|\widetilde{\Theta}\|^2\widetilde{N}^{\bar{q}}(\mathbf{p}) - \sigma_{\bar{q}}\widetilde{\Theta}(\mathbf{p})\big) \qquad \text{pour tout } \mathbf{p} \in \mathcal{L} \text{ dans } S^{\bar{q}} \tag{3.27}$$

$$\widetilde{\delta}_{\mathbf{p}} = \frac{-d\,\sigma_{\bar{q}}\widetilde{\Theta}(\mathbf{p})}{\|\widetilde{N}^{\bar{q}}\|^2\,\|\widetilde{\Theta}\|^2 - \sigma_{\bar{q}}^2} \qquad \text{pour tout autre } \mathbf{p} \in \mathcal{L}. \tag{3.28}$$

Comme dans la section précédente, cette solution est utilisée successivement dans chaque axe. Remarquons en outre que cette solution est explicite : il n'est pas nécessaire de résoudre le système linéaire par une méthode générale, ce qui diminue beaucoup les temps de calculs. Dans le cas optimal d'un système linéaire de taille T avec $O(T)$ éléments non nuls, une méthode de type gradient conjugué trouve la solution en T itérations de coût $O(T)$, c'est à dire en temps $O(T^2)$. Quant à elle, notre solution explicite se calcule en $O(T)$, *i.e.* en temps linéaire par rapport au nombre de coefficients libres.

Résumé d'une déformation de surface composite avec position stricte.

La méthode complète de déformation de surfaces composites avec position stricte prend donc en entrée :

i. Les sous-séquences de nœuds $\widetilde{\mathbf{u}}$ et $\widetilde{\mathbf{v}}$ pour tous les patchs.

ii. Le déplacement $\vec{d}$ pour le point de paramètre $(\bar{u}, \bar{v})$ sur le patch d'indice $\bar{q}$.

iii. Une étendue pour la correction de volume, c'est-à-dire un ensemble $\mathcal{L}$ de points $\mathbf{p}$ libres.

L'algorithme (généralisant l'algorithme 3.3 aux surfaces composites) traite successivement les trois axes :

i. $\widetilde{\delta}X$ est calculé par (3.27) et (3.28) avec $d = d_X$ et $\widetilde{\Theta} = \widetilde{\Theta}_{YZ}$.
δX est calculé par insertion de nœuds, puis $X \leftarrow X + \delta X$.

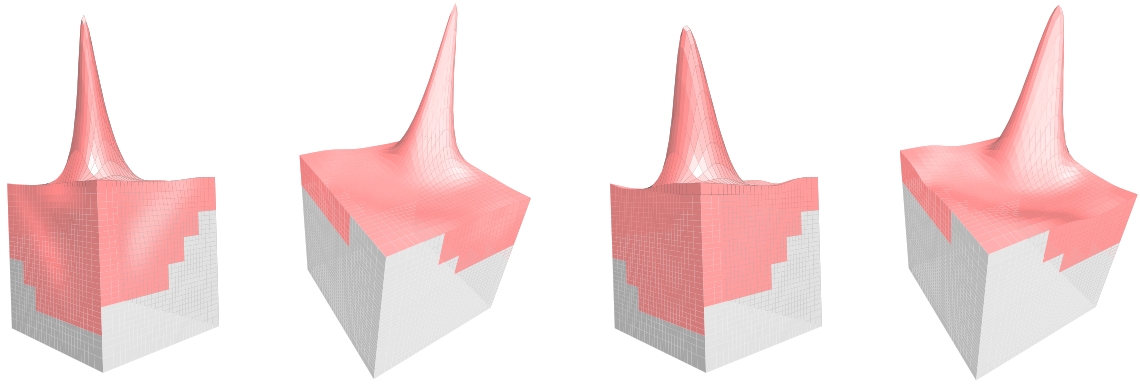
ii. $\widetilde{\delta}Y$ est calculé par (3.27) et (3.28) avec $d = d_Y$ et $\widetilde{\Theta} = \widetilde{\Theta}_{XZ}$.
δY est calculé par insertion de nœuds, puis $Y \leftarrow Y + \delta Y$.

iii. $\widetilde{\delta}Z$ est calculé par (3.27) et (3.28) avec $d = d_Z$ et $\widetilde{\Theta} = \widetilde{\Theta}_{XY}$.
δZ est calculé par insertion de nœuds, puis $Z \leftarrow Z + \delta Z$.

3.2.8 Résultats

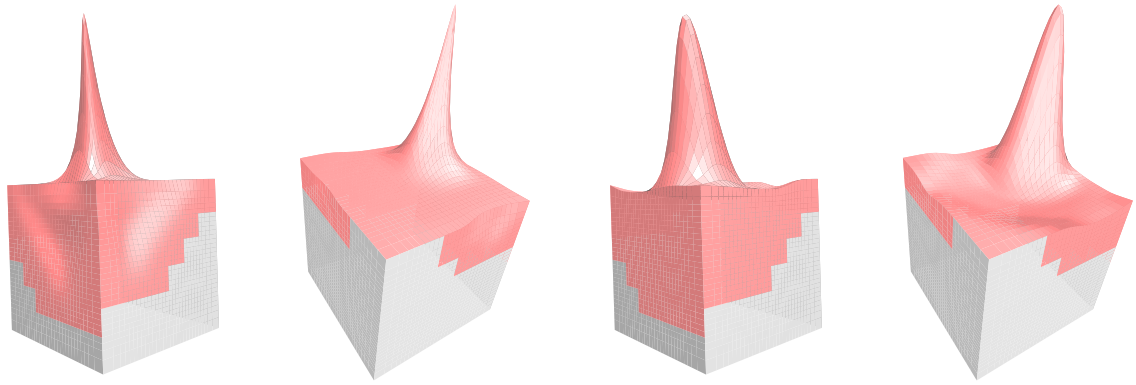
Dans un premier temps, on compare les différentes méthodes de résolution présentées dans les sections 3.2.6 et 3.2.7, à l'aide d'un exemple simple : la figure 3.8 présente une déformation à volume constant d'un cube dont chaque face est un patch. Les coutures sont donc sur les arêtes du cube. Quatre simulations sont présentées :

(a) La répartition équitable du volume avec relaxation de la position est un intermédiaire entre (b) et (c).



(a) Répartition équitable

(b) Répartition proportionnelle à $\vec{d}$



(c) Répartition proportionnelle au déplacement orthogonal

(d) Position stricte

FIG. 3.8 – Comparaison des méthodes non uniformes.
La même déformation est appliquée pour différentes répartitions du volume dans la méthode avec position relaxée (a,b,c) et avec position stricte (d).
Chaque fois deux vues différentes de la même surfaces permettent d'observer les effets de la correction de volume sur les faces latérales (à gauche) et sur la face supérieure (à droite).
La surface initiale est un cube dont chaque face est un patch plan de taille 15×15. L'espace de déformation est obtenu en enlevant un nœud sur deux. La région déformée (en rouge/sombre) est un voisinage de taille 6 en norme 1.

(b) La position est relaxée, et la correction de volume est répartie selon les axes proportionnellement à la même composante du vecteur déplacement. Par conséquent on observe un creux tout autour du pic pour compenser la variation de volume. Ce creux est bien une déformation de la surface dans la même direction que le déplacement, mais dans le sens opposé.

(c) La position est relaxée, et la correction de volume est répartie selon les axes proportionnellement aux composantes orthogonales du vecteur déplacement. La surface n'est pas creusée sur la face supérieure mais sur les faces latérales, et le pic est bien plus fin.

(d) La position est strictement observée : le volume est conservé en même temps que le point est déplacé (méthode section 3.2.7). Le résultat sur cet exemple est très proche du cas (b), ce qui s'explique par le traitement séparé des axes : le voisinage est modifié selon chaque axe pour conserver le volume, en même temps que le déplacement est appliqué sur ce même axe.

Remarquons que les méthodes (b) et (d) ne sont pas pour autant pas équivalentes. La différence tient à ceci :

- pour la méthode (b), on *choisit* de compenser en x une partie du volume proportionnelle à $|d_X|^2$ dans $\|\vec{d}\|^2$.
- pour la méthode (d), est compensée en x la variation de volume *impliquée* par le déplacement $(d_X, 0, 0)$,

ce qui peut être très différent : selon la surface initiale, $|d_X|^2$ peut être prépondérant mais l'impact de $(d_X, 0, 0)$ peut être minime.

Les méthodes avec relaxation (section 3.2.6) semblent donc intéressantes dans la mesure où elles permettent des comportements très différents selon l'objet modélisé (Fig. 3.8(a,b,c)), on peut jouer sur l'effet visuel. Par opposition, la méthode avec position stricte (section 3.2.7, Fig. 3.8(d)) offre moins de libertés mais est plus ergonomique car elle permet de contraindre précisément la position d'un point de la surface. Ceci peut être important, par exemple, si le déplaçement $\vec{d}$ est défini lors d'une collision avec un autre objet : on ne veut pas que les deux objets s'intersectent.

La figure 3.9 présente un exemple plus complexe. La surface initiale (à gauche) représente une figurine de jeu d'échecs vue de face. Elle est constituée de deux patchs de taille 12×26, de degrés 2 en u (qui paramètre la demi-circonférence du cou) et 3 en v (qui paramètre les deux patchs du bas du cou au bout du museau). Ces deux patchs se raccordent sur 3 de leurs bords : au niveau de la bouche, et sur chaque flanc. Un troisième patch ferme la base de la figurine, mais il n'entre pas en jeu dans nos déformations. La déformation est une torsion du museau sur le côté : au milieu sans contrainte, et à droite avec conservation de volume. La méthode utilisée est celle avec position stricte (section 3.2.7), et les séquences de nœuds ont été décimées ($\widetilde{\mathbf{u}}$ et $\widetilde{\mathbf{v}}$) de façon à ne garder qu'un nœud sur 10 transversalement (en u), et un nœud sur 2 longitudinalement (en v). On remarque que grâce à la conservation de volume l'*aspect* de la surface est mieux conservé. On observe surtout le rôle multi-échelle : le museau est déformé largement, mais les détails (protubérance longitudinale) sont conservés et gardent leur position *par rapport* à la forme globale.

La figure 3.10 présente une autre déformation sur la même surface initiale (à gauche) vue de profil. Cette fois-ci le cou est étiré vers l'arrière (un point sur la nuque est déplacé). Sans contrainte (au milieu) le cou gonfle de manière peu réaliste, alors qu'avec conservation du volume

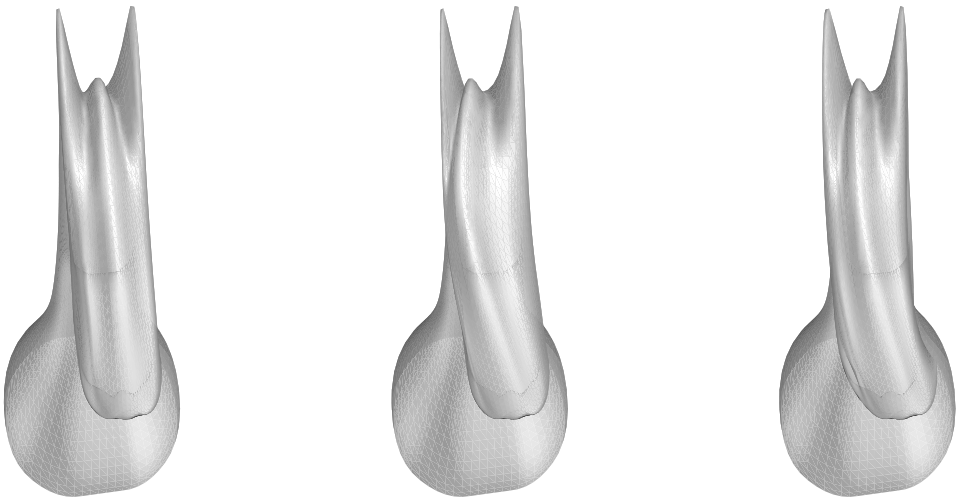


FIG. 3.9 – Torsion du museau sur une figurine de cavalier.
À gauche : la surface initiale ; au milieu : la surface déformée sans contrainte ; à droite : la surface déformée avec conservation de volume selon l'algorithme avec position stricte.

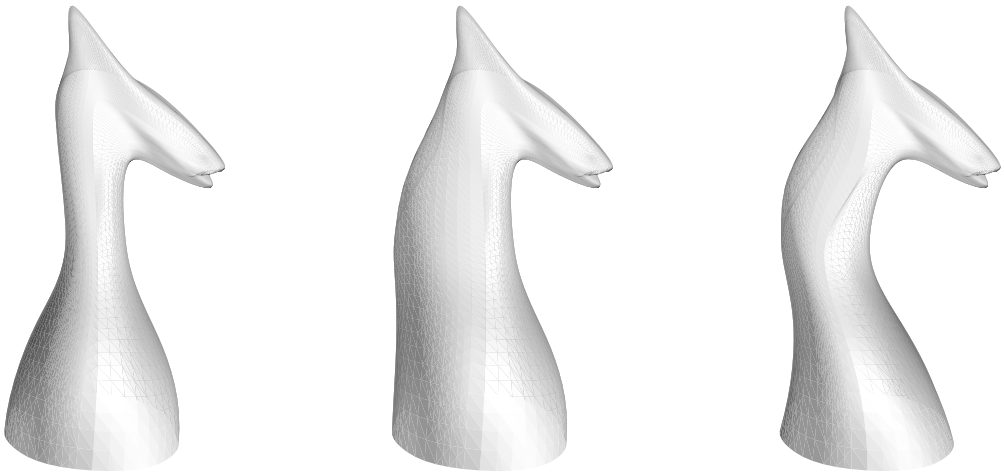


FIG. 3.10 – Déformation du cou sur une figurine de cavalier.
À gauche : la surface initiale ; au milieu : la surface déformée sans contrainte ; à droite : la surface déformée avec conservation de volume selon l'algorithme avec position stricte.

(à droite) le voisinage de la surface se déforme pour un résultat plus convaincant, comme si la figurine était faite de caoutchouc. La région déformée est une bande faisant le tour du cou. Cet exemple montre comment la contrainte géométrique peut imiter un comportement physique qui nous est intuitif.

Notons que ces déformations coûtent quelques centièmes à quelques dixièmes de seconde en temps CPU, hors pré-calculs (chargement de la surface, etc.), sur un PC standard. Il s'en suit que l'on peut considérer cela comme interactif (pour des patchs de cette taille).

Deux principaux efforts nous ont permis d'arriver à ces temps de calcul réduits. D'une part, nous proposons un maximum de pré-calculs, pour ne faire au moment de la déformation, que les calculs strictement nécessaires. D'autre part, nous trouvons une solution explicite des systèmes linéaires, ce qui nous permet de calculer la déformation dans un temps plus réduit que si nous avions dû utiliser une méthode générale de résolution de systèmes creux.

3.3 B-splines uniformes

Dans cette section, nous développons une nouvelle méthode de déformation multirésolution de surfaces sous contrainte de volume. Elle utilise des B-splines uniformes produit tensoriel et s'appuie sur les mêmes principes que la méthode de déformation de courbes présentée section 2.4. Les deux méthodes présentées en section 3.2 remplissent un objectif similaire. Comme nous l'avons fait pour la contrainte d'aire dans la section 2.5, nous nuançons cette nouvelle solution par un apport de possibilités supplémentaires. Il ne s'agit plus seulement de déformation *multi-échelle* mais de déformation *multirésolution*. Nous pouvons non seulement déformer la surface à une échelle quelconque, mais aussi manipuler les détails de la surface grâce à une représentation multirésolution complète. Ceci est rendu possible par l'utilisation de B-splines *uniformes*.

Nous commençons par une présentation (section 3.3.1) de surfaces composites faites de patchs B-spline produit tensoriel, sur lesquels il est possible de construire une analyse multirésolution. Dans la section 3.3.2 nous montrons comment il est possible de calculer le volume englobé par une telle surface. Nous y développons une méthode de calcul quel que soit le niveau de résolution, avec un maximum de pré-calculs pour optimiser les temps de calculs au moment de la déformation. Ceci nous permet de construire une méthode de déformation multirésolution avec conservation du volume (section 3.3.3). Il est possible de contrôler l'aspect et l'étendue de la déformation grâce à la représentation multirésolution. Moyennant des adaptation similaires au cas non uniforme, nous pourrons étendre cette méthode à des surfaces composites (section 3.3.4). Nous illustrons et commentons l'algorithme dans la section 3.3.5.

3.3.1 Surfaces produit tensoriel

Pour créer des surfaces B-splines fermées, il ne suffit pas d'un produit tensoriel de B-splines uniformes unidimensionnelles. En effet, nous avons vu dans le chapitre 2 qu'une courbe *fermée* est obtenue par périodisation sur la séquence de nœuds. Dans le cas des surfaces (voir la section 3.1) il est nécessaire de passer par des surfaces composites, c'est-à-dire définies comme une collection de patchs qui se recollent suivant leurs bords. Les patchs qui constituent la surface sont définis comme produit tensoriel de B-splines uniformes sur un intervalle. La construction de ce modèle est détaillé en annexe A.3, auquel le lecteur pourra se reporter pour plus de détails. Nous en donnons ici les éléments principaux.

Surfaces B-splines.

Soit S une surface composite, décrite par une collection de patchs $\{S^q\}_q$. Chacun de ces patchs est un élément d'un espace V^n construit à partir de deux séquences finies de nœuds

$$\mathbf{u} = \{u_0, u_1, \ldots, u_{m_n+d}\} \quad \text{et} \quad \mathbf{v} = \{v_0, v_1, \ldots, v_{m_n+d}\} \ .$$

Sur ces séquences sont construites des séries de fonctions B-splines unidimensionnelles $\left(N^d_{i_u}(u)\right)_{i_u}$ et $\left(N^d_{i_v}(v)\right)_{i_v}$ de degré d, à partir desquelles nous pouvons exprimer les fonctions d'échelle :

$$\varphi^n_{\mathbf{i}}(u,v) = N^d_{i_u}(u) N^d_{i_v}(v) \quad \text{pour } (0,0) \leq \mathbf{i} = (i_u, i_v) \leq (m_n - 1, m_n - 1) \ . \tag{3.29}$$

Remarquons que les degrés d et les tailles m_n peuvent être différenciés entre u et v, mais par souci de clarté dans les notations nous considérons le même degré d et la même taille m_n dans les deux directions. Ces fonctions d'échelle fines forment une base de l'espace V^n qui va accueillir une analyse multirésolution [SDS96]. Pour cela il est nécessaire que l'espace V^n ait une dimension particulière de la forme $m_n^2 = (d+p2^n)^2$, avec p un entier positif quelconque. Afin que les courbes de bord du patch ne dépendent que des points de bord, les nœuds de bord sont égaux :

$$u_0 = u_1 = \ldots = u_d = 0 \quad \text{et} \quad v_0 = v_1 = \ldots = v_d = 0,$$

$$u_m = \ldots = u_{m+d} = 1 \quad \text{et} \quad v_m = \ldots = v_{m+d} = 1.$$

Pour parler de B-splines *uniformes* il faut que les nœuds intérieurs soient équirépartis sur $[0;1]$:

$$u_{d+i} = v_{d+i} = \frac{i}{p2^n} \text{ pour } 0 \leq i \leq 2^n.$$

Un patch $S^q \in V^n$ s'écrit alors

$$\begin{aligned} S^q \ : \ [0,1]^2 &\longrightarrow \mathbb{R}^3 \\ (u,v) &\longrightarrow S(u,v) = \sum_{\mathbf{i}=(0,0)}^{(m_n-1,m_n-1)} \mathbf{p_i} \ \varphi^n_{\mathbf{i}}(u,v) \ , \end{aligned} \tag{3.30}$$

où $\mathbf{p_i} \in \mathbb{R}^3$ sont les coefficients d'échelle, ou points de contrôle de la surface.

Analyse multirésolution.

Une analyse multirésolution est toujours basée sur une suite d'espaces emboîtés $V^0 \subset V^1 \subset \cdots \subset V^n$. Chaque espace V^j est engendré par des fonctions d'échelle $\varphi^j_{\mathbf{i}}(u,v)$ définies récursivement. Considérons les matrices de fonctions $\varphi^j = \left(\varphi^j_{i_u,i_v}\right)_{i_u,i_v}$. Les fonctions d'échelle grossières sont définies à partir des fonctions d'échelle fines par la relation

$$\varphi^{j-1} \ = \ (P^j)^T \, \varphi^j \, P^j \ ,$$

où la matrice P^j est une matrice dite de reconstruction. Il s'agit en fait d'appliquer le filtre P^j en produit tensoriel sur φ^j. Les fonctions $\varphi^j_{\mathbf{i}}$ s'avèrent être des B-splines sur des sous-séquences de $\mathbf{u}$ et $\mathbf{v}$. L'annexe A.3 fournit l'expression des filtres pour les B-splines quadratiques.

Pour créer l'analyse multirésolution, il faut définir des espaces de détails qui complètent l'espace V^{j-1} dans V^j pour tout $1 \leq j \leq n$. Contrairement au cas unidimensionnel cela implique non pas *un*, mais *trois* espaces de détail W_0^j, W_1^j et W_2^j :

$$V^j \;=\; V^{j-1} \oplus W_0^{j-1} \oplus W_1^{j-1} \oplus W_2^{j-1} \;.$$

Ces espaces sont engendrés par des familles d'ondelettes respectivement notées $\psi_0^{j-1} = \left(\psi_{0,i_u,i_v}^{j-1}\right)_{i_u,i_v}$, ψ_1^{j-1} et ψ_2^{j-1}. Elles sont construites à partir des fonctions d'échelle grâce aux filtres P^j et Q^j :

$$\begin{aligned}
\psi_0^{j-1} &= (Q^j)^T\, \varphi^j\, P^j, \\
\psi_1^{j-1} &= (P^j)^T\, \varphi^j\, Q^j, \\
\psi_2^{j-1} &= (Q^j)^T\, \varphi^j\, Q^j.
\end{aligned}$$

Nous invitons le lecteur à se reporter à l'annexe A.3 pour une explication détaillée de cette construction des espaces de détails. L'ouvrage de Stollnitz *et al* [SDS96] présente aussi un schéma produit tensoriel pour la base de Haar dans son chapitre 3, et évoque le cas des ondelettes B-splines dans l'introduction de son chapitre 10.

En écho à la décomposition des espaces, les coefficients d'échelle $\mathbf{p}^j = \left(\mathbf{p}_{i_u,i_v}^j\right)_{i_u,i_v}$ se décomposent en une série de coefficients d'échelle grossiers $\mathbf{p}^{j-1}$ et trois séries de coefficients d'ondelette $\mathbf{d}_0^{j-1}$, $\mathbf{d}_1^{j-1}$ et $\mathbf{d}_2^{j-1}$ grâce à deux matrices de décomposition A^j et B^j :

$$\begin{aligned}
\mathbf{p}^{j-1} &= A^j\, \mathbf{p}^j\, (A^j)^T \\
\mathbf{d}_0^{j-1} &= B^j\, \mathbf{p}^j\, (A^j)^T \\
\mathbf{d}_1^{j-1} &= A^j\, \mathbf{p}^j\, (B^j)^T \\
\mathbf{d}_2^{j-1} &= B^j\, \mathbf{p}^j\, (B^j)^T \;.
\end{aligned}$$

Ces formules permettent de construire un banc de filtre (voir Fig. A.2 page 142) similaire au cas unidimensionnel (voir Fig. 1.1). Nous sommes ainsi en mesure d'exprimer le patch S^q dans une base multirésolution, à un niveau de résolution $e \in \{0, 1, \ldots, n\}$ quelconque :

$$S^q(u,v) = \sum_{\mathbf{i}} \mathbf{p}_{\mathbf{i}}^e\; \varphi_{\mathbf{i}}^e(u,v) \;+\; \sum_{j=e}^{n-1} \left(\sum_{\mathbf{i}} \mathbf{d}_{0,\mathbf{i}}^j\; \psi_{0,\mathbf{i}}^j(u,v) \;+\; \sum_{\mathbf{i}} \mathbf{d}_{1,\mathbf{i}}^j\; \psi_{1,\mathbf{i}}^j(u,v) \;+\; \sum_{\mathbf{i}} \mathbf{d}_{2,\mathbf{i}}^j\; \psi_{2,\mathbf{i}}^j(u,v) \right) \tag{3.31}$$

qui s'écrit plus synthétiquement :

$$S^q(u,v) = (\mathbf{p}^e)^T \varphi^e \;+\; \sum_{j=e}^{n-1} \left((\mathbf{d}_0^j)^T \psi_0^j \;+\; (\mathbf{d}_1^j)^T \psi_1^j \;+\; (\mathbf{d}_2^j)^T \psi_2^j \right).$$

Réciproquement il est possible de reconstruire les fonctions $\varphi_{\mathbf{i}}^j$ à partir des fonctions d'échelle et des ondelettes du niveau inférieur :

$$\varphi^j \;=\; (A^j)^T\, \varphi^{j-1}\, A^j \;+\; (B^j)^T\, \psi_0^{j-1}\, A^j \;+\; (A^j)^T\, \psi_1^{j-1}\, B^j \;+\; (B^j)^T\, \psi_2^{j-1}\, B^j \;.$$

De même les coefficients $\mathbf{p}^j$ se reconstruisent à partir des coefficients d'échelle $\mathbf{p}^{j-1}$ et des coefficients de détail :

$$\mathbf{p}^j = P^j\, \mathbf{p}^{j-1}\, (P^j)^T \;+\; Q^j\, \mathbf{d}_0^{j-1}\, (P^j)^T \;+\; P^j\, \mathbf{d}_1^{j-1}\, (Q^j)^T \;+\; Q^j\, \mathbf{d}_2^{j-1}\, (Q^j)^T \;.$$

Remarque : il est possible de différencier les niveaux de décomposition e selon u et v, *i.e.* faire une analyse multirésolution anisotrope, mais cela modifie la suite d'espaces d'approximation V^j. Cela est facile à faire d'un point de vue algorithmique, car il suffit de bien déconnecter l'application des filtres entre u et v. Nous l'utilisons d'ailleurs dans les illustrations de ce chapitre. C'est par contre plus lourd au point de vue notations. Pour cette raison, nous ne développons pas ce formalisme anisotrope. Nous demandons au lecteur de bien vouloir accepter cette possibilité sans plus de détails. Il s'avère que cela ne modifie pas significativement les algorithmes finaux que nous allons présenter.

Notation des coefficients multirésolution.

Les coefficients sont codés sous la forme matricielle présentée figure en 3.11. Ces matrices $\mathbf{c}^j \in (\mathbb{R}^3)^{m_n^2}$ sont remplies de coefficients d'échelle et de coefficients d'ondelettes dans $\mathbb{R}^3$ en fonction du niveau $j \in \{0, 1, \ldots, n\}$ de décomposition.

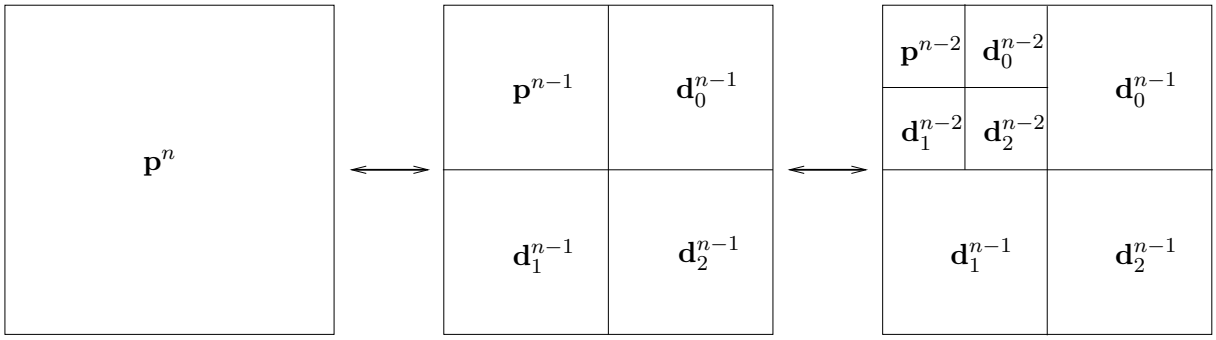


FIG. 3.11 – Stockage des coefficients multirésolution.
De gauche à droite : $\mathbf{c}^n = (X^n, Y^n, Z^n)$, $\mathbf{c}^{n-1} = (X^{n-1}, Y^{n-1}, Z^{n-1})$ et $\mathbf{c}^{n-2} = (X^{n-2}, Y^{n-2}, Z^{n-2})$.

Nous notons chacune de ces matrices $\mathbf{c}^j$ (pour $j = n,\ n-1$ et $n-2$ FIG. 3.11) selon ses composantes (une par axe) X^j, Y^j et Z^j, qui s'écrivent coefficient par coefficient :

$$X^j = \left(x^j_{i_u,i_v}\right)_{i_u,i_v} \in \mathbb{R}^{m_n^2},\ Y^j = \left(y^j_{i_u,i_v}\right)_{i_u,i_v} \in \mathbb{R}^{m_n^2} \quad \text{et} \quad Z^j = \left(z^j_{i_u,i_v}\right)_{i_u,i_v} \in \mathbb{R}^{m_n^2}.$$

Nous attirons l'attention du lecteur sur le fait que la notation $\mathbf{c}^j$ diffère de celle utilisée dans le chapitre précédent : il s'agit ici de *tous* les coefficients multirésolution, coefficients de détails compris.

3.3.2 Calcul du volume

Soit S une surface composite fermée et orientée, faite d'une collection de patchs $S^q(u,v) = (x(u,v), y(u,v), z(u,v))$ définis par (3.30). Dans cette section nous cherchons à calculer le volume inclus dans S en fonction des coefficients de la décomposition. Nous verrons qu'il est possible de l'écrire sous une forme tri-linéaire, dans un premier temps en fonctions des points de contrôle au niveau le plus fin. Nous construisons ensuite une méthode de calcul dans la base multirésolution grâce à l'utilisation des filtres de recomposition P^j et Q^j. Nous pourrons alors envisager des déformations multirésolution de la surface tout en conservant le volume, exprimé dans la base

multirésolution.

Comme cela a déjà été introduit dans la section 3.2.2, le volume $\Theta(S)$ englobé par S peut s'exprimer sous la forme d'une intégrale sur la surface [GOMP98, Elb01]. Dans le cas d'une surface composite, le volume est la somme des intégrales selon chaque patch :

$$\Theta(S) = \sum_{\text{patchs } q} \Theta(S^q)$$

avec

$$\Theta(S^q) = \int_{v=0}^{1}\int_{u=0}^{1} z(u,v)\Big(\frac{\partial x}{\partial u}(u,v)\frac{\partial y}{\partial v}(u,v) - \frac{\partial x}{\partial v}(u,v)\frac{\partial y}{\partial u}(u,v)\Big)\,du\,dv\ . \tag{3.32}$$

Formulation du volume d'un patch dans la base fine.

Notre objectif est d'écrire ce volume en fonction des points de contrôle qui définissent la surface. En injectant l'égalité (3.30) dans l'équation (3.32), le volume du patch s'écrit sous forme tri-linéaire par rapport aux composantes des points :

$$\Theta(S^q) = \sum_{\mathbf{i},\mathbf{j},\mathbf{k}} x^n_{\mathbf{i}}\, y^n_{\mathbf{j}}\, z^n_{\mathbf{k}}\, \theta^n_{\mathbf{i}\,\mathbf{j}\,\mathbf{k}} \quad \text{pour} \quad (0,0) \le \mathbf{i},\mathbf{j},\mathbf{k} \le (m_n-1, m_n-1) \tag{3.33}$$

où

$$\theta^n_{\mathbf{i}\,\mathbf{j}\,\mathbf{k}} = \iint \varphi^n_{\mathbf{k}}\Big(\frac{\partial\varphi^n_{\mathbf{i}}}{\partial u}\frac{\partial\varphi^n_{\mathbf{j}}}{\partial v} - \frac{\partial\varphi^n_{\mathbf{i}}}{\partial v}\frac{\partial\varphi^n_{\mathbf{j}}}{\partial u}\Big) \quad \in \mathbb{R}.$$

Dans la mesure où les fonctions d'échelle intervenant dans cette intégrale sont des produits de B-splines (voir équation (3.29)), il est possible de calculer les $\theta^n_{\mathbf{i}\,\mathbf{j}\,\mathbf{k}}$ par intégration formelle. Néanmoins, rappelons qu'il nous faudra calculer le volume (3.33) au cours d'un processus d'édition. Il serait donc trop coûteux *en temps* de calculer ces intégrales à la volée. Il serait à l'opposé trop coûteux *en mémoire* de stocker les m_n^6 valeurs $\theta^n_{\mathbf{i}\,\mathbf{j}\,\mathbf{k}}$. Nous allons donc opter pour une solution intermédiaire. En utilisant la définition (3.29), il est possible de séparer les intégrales en u et en v :

$$\begin{aligned}
\theta^n_{\mathbf{i}\,\mathbf{j}\,\mathbf{k}} &= \int_0^1 N'_{i_u}(u)N_{j_u}(u)N_{k_u}(u)\,du \int_0^1 N_{i_v}(v)N'_{j_v}(v)N_{k_v}(v)\,dv \\
&\quad - \int_0^1 N_{i_u}(u)N'_{j_u}(u)N_{k_u}(u)\,du \int_0^1 N'_{i_v}(v)N_{j_v}(v)N_{k_v}(v)\,dv \\
&= \theta^{n,u}_{j_u k_u i_u}\theta^{n,v}_{i_v k_v j_v} - \theta^{n,u}_{i_u k_u j_u}\theta^{n,v}_{j_v k_v i_v}
\end{aligned}$$

avec

$$\theta^{n,u}_{ijk} = \theta^{n,u}_{jik} = \int_0^1 N_{i_u}(u)\,N_{j_u}(u)\,N'_{k_u}(u)\,du \tag{3.34}$$

$$\theta^{n,v}_{ijk} = \theta^{n,v}_{jik} = \int_0^1 N_{i_v}(v)\,N_{j_v}(v)\,N'_{k_v}(v)\,dv \tag{3.35}$$

Il devient alors possible de calculer une seule fois et de stocker les m_n^3 coefficients $\theta^{n,u}_{ijk}$ et $\theta^{n,v}_{ijk}$. Le calcul du volume par (3.33) se fait alors en temps $O(m_n^6)$. Rappelons qu'un patch S^q possède m_n^2 points de contrôle, donc le coût est cubique.

Formulation du volume d'un patch dans la base multirésolution.

Pour effectuer une édition multirésolution de la surface tout en contrôlant son volume, ces formules ne suffisent pas : nous devons exprimer le volume dans la base multirésolution. Il nous faut donc utiliser la formulation multirésolution (3.31). Formellement, il s'agit d'injecter cette équation (3.31) dans la formulation intégrale (3.32) pour écrire le volume sous forme tri-linéaire. Soit e le niveau de décomposition du patch. Le volume se formule en fonction des composantes des coefficients multirésolution à toutes les échelles :

$$\Theta(S^q) = \sum_{\mathbf{i},\mathbf{j},\mathbf{k}} x^e_{\mathbf{i}}\, y^e_{\mathbf{j}}\, z^e_{\mathbf{k}}\, \theta^e_{\mathbf{i}\,\mathbf{j}\,\mathbf{k}} \quad \text{pour} \quad (0,0) \leq \mathbf{i},\mathbf{j},\mathbf{k} \leq (m_n - 1, m_n - 1) \tag{3.36}$$

où

$$\theta^e_{\mathbf{i}\,\mathbf{j}\,\mathbf{k}} = \theta^{e,u}_{j_u k_u i_u}\, \theta^{e,v}_{i_v k_v j_v} - \theta^{e,u}_{i_u k_u j_u}\, \theta^{e,v}_{j_v k_v i_v} \ .$$

L'écriture de ces coefficients sous forme d'intégrales similaires à (3.34) et (3.35) nécessite des notations un peu compliquées. Quant à lui, le calcul sous forme intégrale est non trivial, car il fait intervenir des ondelettes de tous niveaux et des fonctions échelles, parfois dérivées. Afin de calculer ces coefficients, nous allons plutôt repartir de l'expression tri-linéaire (3.33) et travailler avec les filtres de synthèse. Nous donnons ici les grandes lignes de cette méthode, qui est l'exacte généralisation de celle développée pour le calcul de l'aire des courbes en base multirésolution présenté en section 2.3. Le lecteur se reportera à cette dernière pour de plus amples détails.

Nous proposons donc une méthode de calcul récursif des $\theta^{e,u}_{i_u j_u k_u}$ et $\theta^{e,v}_{i_v j_v k_v}$ respectivement à partir des $\theta^{n,u}_{i_u j_u k_u}$ et $\theta^{n,v}_{i_v j_v k_v}$ définis par les intégrales (3.34) et (3.35). Il s'agit d'une récursion descendante sur $j = n, n-1, \ldots, e$.

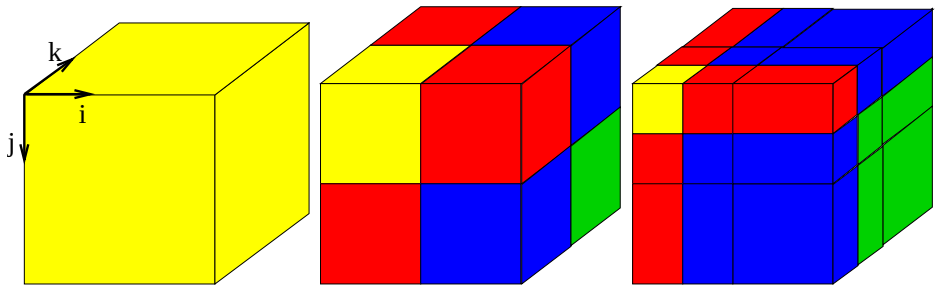


FIG. 3.12 – Calcul récursif des matrices 3D de volume.
De gauche à droite : θ^n, θ^{n-1} et θ^{n-2}.

Notons indifféremment θ^j les matrices $\left(\theta^{j,u}_{i_u j_u k_u}\right)_{i_u j_u k_u}$ et $\left(\theta^{j,v}_{i_v j_v k_v}\right)_{i_v j_v k_v}$ pour $j \in \{0, 1, \ldots, n\}$. Il s'agit de matrices tridimensionnelles de taille $m_n \times m_n \times m_n$. Il est alors possible de calculer θ^{j-1} à partir de θ^j en appliquant les filtres transposés $(P^j)^T$ et $(Q^j)^T$ sous forme produit tensoriel sur des sous-matrices. La figure 3.12 illustre ce processus sur le début de la récursion $\theta^n \rightarrow \theta^{n-1} \rightarrow \theta^{n-2}$:

- Les blocs *verts* (en bas, au fond, à droite) sont invariants.
- Sur les blocs *bleus* (sombres) les filtres $(P^j)^T$ et $(Q^j)^T$ sont appliqués selon une dimension seulement (résultant en un nouveau bloc bleu et un bloc vert).
- Sur les blocs *rouges* (allongés suivant les arêtes) les filtres sont appliqués selon deux dimensions (résultant en un nouveau bloc rouge, deux blocs bleus et un bloc vert).

– Sur le bloc *jaune* (devant, en haut, à gauche) les filtres sont appliqués selon les trois dimensions (résultant en un nouveau bloc jaune, trois blocs rouges, trois blocs bleus et un bloc vert).

Résumé du calcul de volume d'un patch.

Nous venons de développer une méthode de calcul du volume en base multirésolution pour un niveau de résolution e quelconque (c'est-à-dire pour un patch défini par (3.31)) constitué de trois étapes :

i. Calcul et stockage des $\theta^{n,u}_{i_u j_u k_u}$ et $\theta^{n,v}_{i_v j_v k_v}$ par les formules intégrales (3.34) et (3.35).

ii. Calcul et stockage des $\theta^{e,u}_{i_u j_u k_u}$ et $\theta^{e,v}_{i_v j_v k_v}$ par la récursion $\theta^n \to \theta^{n-1} \to \cdots \to \theta^e$.

iii. Calcul du volume par la formule (3.36).

La première étape est un pré-calcul : il est effectué au moment du chargement de la surface. La deuxième est effectuée pour chaque changement du niveau de résolution dans l'éditeur. Seule la troisième étape est effectuée à chaque déformation de la surface, ce qui limite les temps de calculs.

Pour la manipulation de surfaces composites ces trois étapes s'appliquent à chacun des patchs qui la constituent.

3.3.3 Déformation à volume constant

Soit S^q un patch uniforme produit tensoriel multirésolution que l'on souhaite déformer en conservant son volume. Nous construisons ici une boucle d'édition du type Fig. 1.5(b) qui permet d'atteindre cet objectif. Il s'agit donc d'une étape d'édition suivie d'une étape de correction du volume.

Après avoir défini le niveau de résolution e auquel il souhaite éditer S^q, l'utilisateur édite un point $\mathbf{p}^e_{\mathbf{i}}$ du maillage de contrôle. Le patch ainsi déformé s'exprime sous la forme (3.31). Il est défini par un ensemble de coefficients d'échelle et d'ondelette de $\mathbb{R}^3$ regroupés dans une matrice $\bar{\mathbf{c}}^e = (\bar{X}^e, \bar{Y}^e, \bar{Z}^e) \in (\mathbb{R}^{m_n^2})^3$ (voir Fig. 3.11 page 68). Son volume $\bar{\Theta} = \Theta(\bar{X}^e, \bar{Y}^e, \bar{Z}^e)$ peut être calculé par la formule (3.36), et est *a priori* différent du volume de référence Θ_{ref} que l'on souhaite qu'il ait.

Correction globale du volume.

Il nous faut définir une correction $\delta\mathbf{c}^e = (\delta X^e, \delta Y^e, \delta Z^e) \in (\mathbb{R}^{m_n^2})^3$ de façon à définir une surface finale $\bar{\mathbf{c}}^e + \delta\mathbf{c}^e = (X^e, Y^e, Z^e)$ qui :

i. est la plus proche possible de la surface déformée $\bar{\mathbf{c}}^e$, et

ii. a pour volume Θ_{ref}.

À l'instar de la méthode pour les courbes (section 2.4), nous atteignons cet objectif grâce à une minimisation sous contrainte. La fonction objectif sert à assurer la proximité (i), on considère la norme quadratique de la déformation :

$$\|\delta\mathbf{c}^e\|^2 = \|\delta X^e\|^2 + \|\delta Y^e\|^2 + \|\delta Z^e\|^2$$

Nous ne reprenons pas l'énergie de tension $\mathcal{E}$ utilisée pour les courbes, car, comme nous l'avons souligné dans les commentaires de la section 2.5.2, celle-ci n'est pas d'une grande aide.

De plus nous verrons que nous pouvons ainsi exprimer le résultat explicitement, c'est-à-dire que nous n'aurons pas besoin d'une méthode générale de résolution de systèmes creux, donc le processus sera plus rapide.
Pour satisfaire la condition de volume (ii), nous ajoutons une contrainte à la minimisation :

$$\Theta(X^e, Y^e, Z^e) = \Theta(\bar{X}^e + \delta X^e, \bar{Y}^e + \delta Y^e, \bar{Z}^e + \delta Z^e) = \Theta_{ref} \ , \tag{3.37}$$

signifiant que le volume $\Theta(X^e, Y^e, Z^e)$ de la surface finale doit être égal au volume de référence.

Il s'agit donc de minimiser une fonction objectif quadratique sous une contrainte cubique. Cela serait trop coûteux en temps si l'on souhaite appliquer des déformations en temps réel. Par conséquent nous allons linéariser cette contrainte en approximant le volume du patch final. En utilisant la multi-linéarité de la fonction de volume Θ, nous pouvons écrire :

$$\begin{aligned}\Theta(X^e, Y^e, Z^e) \ &= \ \Theta(\bar{X}^e, \bar{Y}^e, \bar{Z}^e) + \Theta(\bar{X}^e, \bar{Y}^e, \delta Z^e) + \Theta(\bar{X}^e, \delta Y^e, \bar{Z}^e) + \Theta(\delta X^e, \bar{Y}^e, \bar{Z}^e) \\ &+ \ \Theta(\bar{X}^e, \delta Y^e, \delta Z^e) + \Theta(\delta X^e, \bar{Y}^e, \delta Z^e) + \Theta(\delta X^e, \delta Y^e, \bar{Z}^e) + \Theta(\delta X^e, \delta Y^e, \delta Z^e).\end{aligned}$$

Le premier terme est constant, et les trois suivants sont linéaires par rapport aux variables δX^e, δY^e et δZ^e. En supposant que la correction est petite par rapport à la dimension du patch, on néglige les 4 derniers termes qui sont d'ordre 2 ou 3. Alors le volume de S^q est approximé par

$$\Theta(X^e, Y^e, Z^e) \approx \Theta(\bar{X}^e, \bar{Y}^e, \bar{Z}^e) + \Theta(\bar{X}^e, \bar{Y}^e, \delta Z^e) + \Theta(\bar{X}^e, \delta Y^e, \bar{Z}^e) + \Theta(\delta X^e, \bar{Y}^e, \bar{Z}^e) \ .$$

Ainsi la contrainte (3.37) peut être approximée par la contrainte linéaire :

$$\Theta(\bar{X}^e, \bar{Y}^e, \delta Z^e) + \Theta(\bar{X}^e, \delta Y^e, \bar{Z}^e) + \Theta(\delta X^e, \bar{Y}^e, \bar{Z}^e) = \Theta_{ref} - \bar{\Theta} \ . \tag{3.38}$$

Le problème se reformule alors :

$$\min \ \|\delta X^e\|^2 + \|\delta Y^e\|^2 + \|\delta Z^e\|^2$$

$$\text{sous contrainte} \quad \Theta(\bar{X}^e, \bar{Y}^e, \delta Z^e) + \Theta(\bar{X}^e, \delta Y^e, \bar{Z}^e) + \Theta(\delta X^e, \bar{Y}^e, \bar{Z}^e) = \Theta_{ref} - \bar{\Theta}.$$

Correction locale du volume grâce à une minimisation partielle.

Jusque là nous avons formulé une correction globale, c'est-à-dire que tous les coefficients $\mathbf{c}_\mathbf{i}^e$ sont modifiés. Ces coefficients regroupent des coefficients d'échelle $\mathbf{p}_\mathbf{i}^e$ et des coefficients d'ondelette $\mathbf{d}_{0,\mathbf{i}}^j$, $\mathbf{d}_{1,\mathbf{i}}^j$ et $\mathbf{d}_{2,\mathbf{i}}^j$ pour tous les niveaux $j = e, e+1, \ldots, n-1$. Or il est souhaitable de ne modifier que certains de ces coefficients pour mieux contrôler la déformation et tirer profit de la représentation multirésolution :

- Fixer le point de contrôle édité par l'utilisateur permet de mieux respecter la déformation initiale, et d'augmenter l'ergonomie de l'outil d'édition.
- Fixer éventuellement les coefficients de détails permet de conserver les détails caractéristiques de la surface en ne corrigeant que la forme globale.
- Ne laisser libres que les coefficients correspondant à une certaine région de la surface permet de localiser la déformation sur cette région.

Pour faire cela nous allons effectuer la minimisation sur uniquement une partie $\mathcal{L}$ des coefficients. Les coefficients $\mathbf{c} \in \bar{\mathbf{c}}^e$ sont donc répartis en deux catégories : ceux qui sont fixés (*i.e.* la déformation associée $\delta_\mathbf{c} = 0$), et ceux qui sont libres (*i.e.* $\delta_\mathbf{c}$ entre en jeu dans la minimisation). L'ensemble des coefficients libres est noté $\mathcal{L}$. Notre objectif est de simplement réécrire les termes

de la minimisation en fonction de cette répartition entre points fixés et points libres, au lieu de l'écriture en fonction des trois axes. Autrement dit nous allons décomposer la fonction objectif $\|\delta\mathbf{c}^e\|^2$ et la contrainte (3.38) comme sommes sur tous les coefficients libres.

Définissons quelques notations plus appropriées à cette répartition. Définissons $\Theta^q_{\bar{Y}^e\bar{Z}^e} = \big(\Theta^q_{\bar{Y}^e\bar{Z}^e}(\mathbf{i})\big)_{\mathbf{i}}$ comme (3.10) par

$$\Theta^q_{\bar{Y}^e\bar{Z}^e}(\mathbf{i}) = \sum_{\mathbf{j},\mathbf{k}} \theta^e_{\mathbf{i}\,\mathbf{j}\,\mathbf{k}}\, \bar{y}^e_{\mathbf{j}}\, \bar{z}^e_{\mathbf{k}} \quad \text{pour} \quad (0,0) \le \mathbf{i} \le (m^n-1, m^n-1)$$

ainsi que $\Theta^q_{\bar{X}^e\bar{Z}^e}$ et $\Theta^q_{\bar{X}^e\bar{Y}^e}$ symétriquement.
Par un léger abus de notations, nous utilisons le coefficient $\mathbf{c}$ pour indexer les vecteurs, à la place de l'indice $\mathbf{i}$ qui le référence. Alors, à chaque coefficient $\mathbf{c} \in \mathcal{L}$ nous associons une correction $\delta_{\mathbf{c}} \in \mathbb{R}^3$. La variation de volume induite par la correction $\delta_{\mathbf{c}}$ d'*un seul* coefficient $\mathbf{c}$ libre peut s'écrire sous forme d'un produit scalaire de $\mathbb{R}^3$

$$\Theta^q(\mathbf{c}) \cdot \delta_{\mathbf{c}}\ ,\ \mathbf{c} \in \mathcal{L},$$

où

$$\Theta^q(\mathbf{c}) = \begin{pmatrix} \Theta^q_{\bar{Y}^e\bar{Z}^e}(\mathbf{c}) \\ \Theta^q_{\bar{X}^e\bar{Z}^e}(\mathbf{c}) \\ \Theta^q_{\bar{X}^e\bar{Y}^e}(\mathbf{c}) \end{pmatrix} \in \mathbb{R}^3\ , \tag{3.39}$$

et donc la contrainte linéarisée (3.38) se reformule

$$\sum_{\mathbf{c}\in\mathcal{L}} \Theta^q(\mathbf{c}) \cdot \delta_{\mathbf{c}} = \Theta_{ref} - \bar{\Theta}\ .$$

En décomposant la norme quadratique $\|\delta\mathbf{c}^e\|^2$ sous forme d'une somme $\sum \|\delta_{\mathbf{c}}\|^2$ sur tous les coefficients libres, la minimisation partielle s'écrit enfin

$$\min \sum_{\mathbf{c}\in\mathcal{L}} \|\delta_{\mathbf{c}}\|^2 \quad \text{sous contrainte} \quad \sum_{\mathbf{c}\in\mathcal{L}} \Theta^q(\mathbf{c}) \cdot \delta_{\mathbf{c}} = \Theta_{ref} - \bar{\Theta}. \tag{3.40}$$

En utilisant un multiplicateur de Lagrange λ elle se reformule à l'aide du lagrangien

$$g(\mathbf{c}^e, \lambda) = \sum_{\mathbf{c}\in\mathcal{L}} \|\delta_{\mathbf{c}}\|^2 \ +\ \lambda\big(\sum_{\mathbf{c}\in\mathcal{L}} \Theta^q(\mathbf{c}) \cdot \delta_{\mathbf{c}} - (\Theta_{ref} - \bar{\Theta})\big)$$

et la solution à (3.40) est un point selle de ce lagrangien :

$$\vec{\nabla} g = \vec{0} \iff \begin{cases} 2\delta_{\mathbf{c}} + \lambda\Theta^q(\mathbf{c}) &= \vec{0} & \text{pour tout } \mathbf{c} \in \mathcal{L} \\ \sum \Theta^q(\mathbf{c}) \cdot \delta_{\mathbf{c}} &= \Theta_{ref} - \bar{\Theta} & \text{somme sur les } \mathbf{c} \in \mathcal{L}. \end{cases}$$

Ce système linéaire a pour solution :

$$\lambda = \frac{-2(\Theta_{ref} - \bar{\Theta})}{\|\Theta^q\|^2}$$

et

$$\delta_{\mathbf{c}} = \frac{\Theta_{ref} - \bar{\Theta}}{\|\Theta^q\|^2}\, \Theta^q(\mathbf{c}), \quad \text{pour } \mathbf{c} \in \mathcal{L}, \tag{3.41}$$

avec le coefficient de normalisation

$$\|\Theta^q\|^2 = \sum_{\mathbf{c}\in\mathcal{L}} \|\Theta^q(\mathbf{c})\|^2.$$

La formule (3.41) nous fournit donc une déformation locale pour corriger le volume du patch de $\bar{\Theta}$ à Θ_{ref}. Pour la rendre plus utilisable, il nous faut étendre cette solution aux surfaces composites.

3.3.4 Surfaces composites

Soit $S = \{S^q\}_q$ une surface composite que l'on souhaite déformer à volume constant. Nous allons étendre la méthode présentée ci-dessus pour un patch unique, avec les mêmes idées que dans les sections 3.2.5, 3.2.6 et 3.2.7 qui généralisent les sections 3.2.3 et 3.2.4. Comme en section 3.3.3, supposons que :

i. L'utilisateur a choisi un niveau de résolution e pour l'édition. Tous les patchs S^q constituant la surface sont donc représentés par leurs coefficients $\mathbf{c}^e$ au niveau e. Le problème de la cohérence des séquences de nœuds entre les patchs (présenté section 3.2.5 pour les B-splines non uniformes) est plus simple dans le cadre uniforme présent. Une séquence uniforme est toujours symétrique. Il suffit de s'assurer que le degré, le nombre de points de contrôle et le niveau de résolution est homogène entre les deux patchs qui forment une couture.

ii. L'utilisateur a sélectionné et édité un point de contrôle $\mathbf{p}_\mathbf{i}^e$ sur un des patchs. Comme pour le cas de la conservation d'aire pour courbes B-splines uniformes (section 2.4), nous avons choisi de permettre d'éditer un point de contrôle plutôt qu'un point de la surface, comme cela est le cas dans les sections 3.2.6 et 3.2.7. Cette alternative est discutée dans la section 3.4. Lors de cette édition, la surface est déformée à l'échelle grossière e, et la surface $\bar{S}$ obtenue a pour volume $\bar{\Theta}$, qui est a priori différent du volume de référence Θ_{ref}. Le volume $\bar{\Theta}$ peut être calculé en sommant le volume des patchs $\Theta(S^q)$ (voir section 3.3.2) qui s'expriment chacun dans la base multirésolution par la formule tri-linéaire (3.36).

iii. L'utilisateur a défini l'ensemble $\mathcal{L}$ des coefficients qu'il souhaite voir modifiés (coefficients *libres*) lors de la correction de volume, de façon à obtenir une surface finale de volume Θ_{ref}. Pour définir ces coefficients nous proposons de :
- Définir une zone d'influence sur la surface qui entoure le point édité. Elle est définie par une distance sur le maillage et un certain seuil (voir section 3.2.5).
- Choisir si le point édité est libre ou fixé (voir section 2.4.4). Ceci permet à l'utilisateur de contrôler plus finement sa déformation.
- Choisir de laisser libres ou pas les coefficients de détail. Nous avons déjà discuté cette liberté dans la section 2.4.4 : le fait de fixer les détails permet de ne modifier que la forme globale de la surface, sans affecter les détails qui la caractérisent. De plus cela réduit énormément la taille du système linéaire.

Dès lors, on définit une déformation $\delta_{\mathbf{c}} \in \mathbb{R}^3$ pour chacun des coefficients libres. Pour cela nous minimisons de nouveau la norme quadratique de la déformation :

$$\sum_{\mathbf{c}\in\mathcal{L}} \|\delta_{\mathbf{c}}\|^2$$

sous contrainte de la correction de volume. Chaque $\delta_{\mathbf{c}}$ a une influence sur le volume du type $\Theta^q(\mathbf{c}) \cdot \delta_{\mathbf{c}}$, mais pour tous les patchs q sur lesquels le coefficient $\mathbf{c}$ a une influence :

- Si $\mathbf{c}$ est un coefficient de détail, il n'influence que le patch auquel il se rattache.
- Si $\mathbf{c}$ est un point de contrôle à l'intérieur d'un patch, il n'influence que ce patch.
- Si $\mathbf{c}$ est un point de contrôle sur une couture, il influence les deux patchs qui ont cette couture en commun.
- Si $\mathbf{c}$ est un point de contrôle sur un angle entre plusieurs patchs, il influence tous ces patchs.

L'impact global de $\delta_{\mathbf{c}}$ sur le volume s'écrit avec la somme des $\Theta^q(\mathbf{c})$ (voir Eq. (3.39)) pour tous les patchs q que $\mathbf{c}$ influence :

$$\sum_{q\,/\,\mathbf{c}\in S^q} \Theta^q(\mathbf{c}) \cdot \delta_{\mathbf{c}} \ .$$

Par conséquent la contrainte de correction de volume est

$$\sum_{\mathbf{c}\in\mathcal{L}} \sum_{q\,/\,\mathbf{c}\in S^q} \Theta^q(\mathbf{c}) \cdot \delta_{\mathbf{c}} = \Theta_{ref} - \bar{\Theta} \ .$$

Notre problème se reformule

$$\min \sum_{\mathbf{c}\in\mathcal{L}} \|\delta_{\mathbf{c}}\|^2 \quad \text{sous contrainte} \quad \sum_{\mathbf{c}\in\mathcal{L}} \sum_{q\,/\,\mathbf{c}\in S^q} \Theta^q(\mathbf{c}) \cdot \delta_{\mathbf{c}} = \Theta_{ref} - \bar{\Theta}$$

dont la solution est un point stationnaire du lagrangien

$$g(\delta, \lambda) \ = \sum_{\mathbf{c}\in\mathcal{L}} \|\delta_{\mathbf{c}}\|^2 \ + \ \lambda\Big(\sum_{\mathbf{c}\in\mathcal{L}} \sum_{q\,/\,\mathbf{c}\in S^q} \Theta^q(\mathbf{c}) \cdot \delta_{\mathbf{c}} - (\Theta_{ref} - \bar{\Theta}) \Big),$$

c'est-à-dire

$$\vec{\bigtriangledown} g = \vec{0} \quad \Longleftrightarrow \begin{cases} 2\delta_{\mathbf{c}} + \lambda \sum_q \Theta^q(\mathbf{c}) \ = \ \vec{0} & \text{pour tout } \mathbf{c} \in \mathcal{L} \\ \sum_{\mathbf{c}} \sum_q \Theta^q(\mathbf{c}) \cdot \delta_{\mathbf{c}} \ = \ \Theta_{ref} - \bar{\Theta} & \text{somme sur les } \mathbf{c} \in \mathcal{L}. \end{cases}$$

Ce système est très creux, et sa solution peut être explicitée par :

$$\lambda = \frac{-2(\Theta_{ref} - \bar{\Theta})}{\|\Theta\|^2}$$

et

$$\delta_{\mathbf{c}} = \frac{\Theta_{ref} - \bar{\Theta}}{\|\Theta\|^2} \ \Theta(\mathbf{c}) \quad \text{pour tout coefficient } \mathbf{c} \in \mathcal{L}, \tag{3.42}$$

avec le coefficient de normalisation

$$\|\Theta\|^2 = \sum_{\mathbf{c}\in\mathcal{L}} \left\| \sum_{q\,/\,\mathbf{c}\in S^q} \Theta^q(\mathbf{c}) \right\|^2 .$$

Résumé du processus de déformation de surfaces composites.

Nous avons maintenant tous les outils pour constituer un processus de déformation de surfaces composites uniformes à volume constant :

i. L'utilisateur choisit un niveau de résolution e pour l'édition. Tous les patchs sont décomposés à ce niveau.

ii. L'utilisateur sélectionne et édite un point de contrôle $\mathbf{p}_\mathbf{i}^e$ sur un des patchs, typiquement par drag-and-drop dans un éditeur 3D. Cela modifie uniquement ce patch, sur une étendue qui dépend du niveau de résolution e.

iii. L'utilisateur définit un ensemble $\mathcal{L}$ de coefficients multirésolution libres pour corriger le volume, par exemple en utilisant les options discutées au début de cette section.

iv. Une surface finale est calculée dans sa décomposition multirésolution grâce à une déformation définie par (3.42).

Pour construire un éditeur de surface ergonomique, il s'agit en pratique de décomposer la surface une fois pour toutes (i). Ensuite l'utilisateur spécifie ses critères pour définir les coefficients libres (iii). Dès qu'il sélectionne un point de contrôle chaque déplacement de souris est enregistré comme édition (ii) et la surface est automatiquement corrigée (iii) avant de traiter le nouveau déplacement.

Le code que nous avons développé n'utilise pas une telle interface graphique, mais prend en entrée un fichier de commandes qui déclenche des déformations, des opérations d'entrée-sortie, et permet de modifier les paramètres (niveau de résolution, étendue de la déformation, etc.) au cours d'une édition complexe.

Grâce à la résolution explicite (3.42) du système linéaire, et grâce à tous les éléments qui sont pré-calculés et stockés avant la déformation, les déformations peuvent être calculées en temps réel.

3.3.5 Résultats

Nous analysons notre méthode à partir de la figure 3.13, qui regroupe tous les aspects importants. Deux figurines d'échecs (cavalier sans crinière en haut, avec crinière en bas) sont représentées de gauche à droite sous leur forme initiale, déformées sans contrainte, et déformées avec conservation de volume. Ces surfaces sont constituées de deux patchs principaux (le dos et le dessus de la tête pour l'un, le devant et le dessous de la tête pour l'autre), en plus d'un patch fermant le pied de la figurine. Ces surfaces (similaires aux figures 3.9 et 3.10) sont déformées identiquement par déplacement d'un point de contrôle grossier (à l'arrière de l'encolure) vers l'avant, ce qui équivaut à un déplacement de la tête vers l'arrière. Sans conservation de volume (au milieu) le cheval est comme étranglé, son cou s'affine. La conservation de volume (à droite) apporte une nette amélioration du point de vue du réalisme. Vu que la déformation est relativement locale (une bande autour du cou), la correction de volume ne se répercute pas sur les parties éloignées (la tête).

Nous remarquons ici que même petite, une déformation seule peut significativement détériorer le réalisme de la représentation. En outre, comme nous l'avons déjà souligné, notre modèle n'est bien adapté qu'aux *petites* déformations (voir à ce sujet la linéarisation du volume). Pour des déformations plus importantes il faut envisager plusieurs petites déformations successives. L'interface que nous proposons (manipulation des points de contrôles) n'est alors pas propice à de

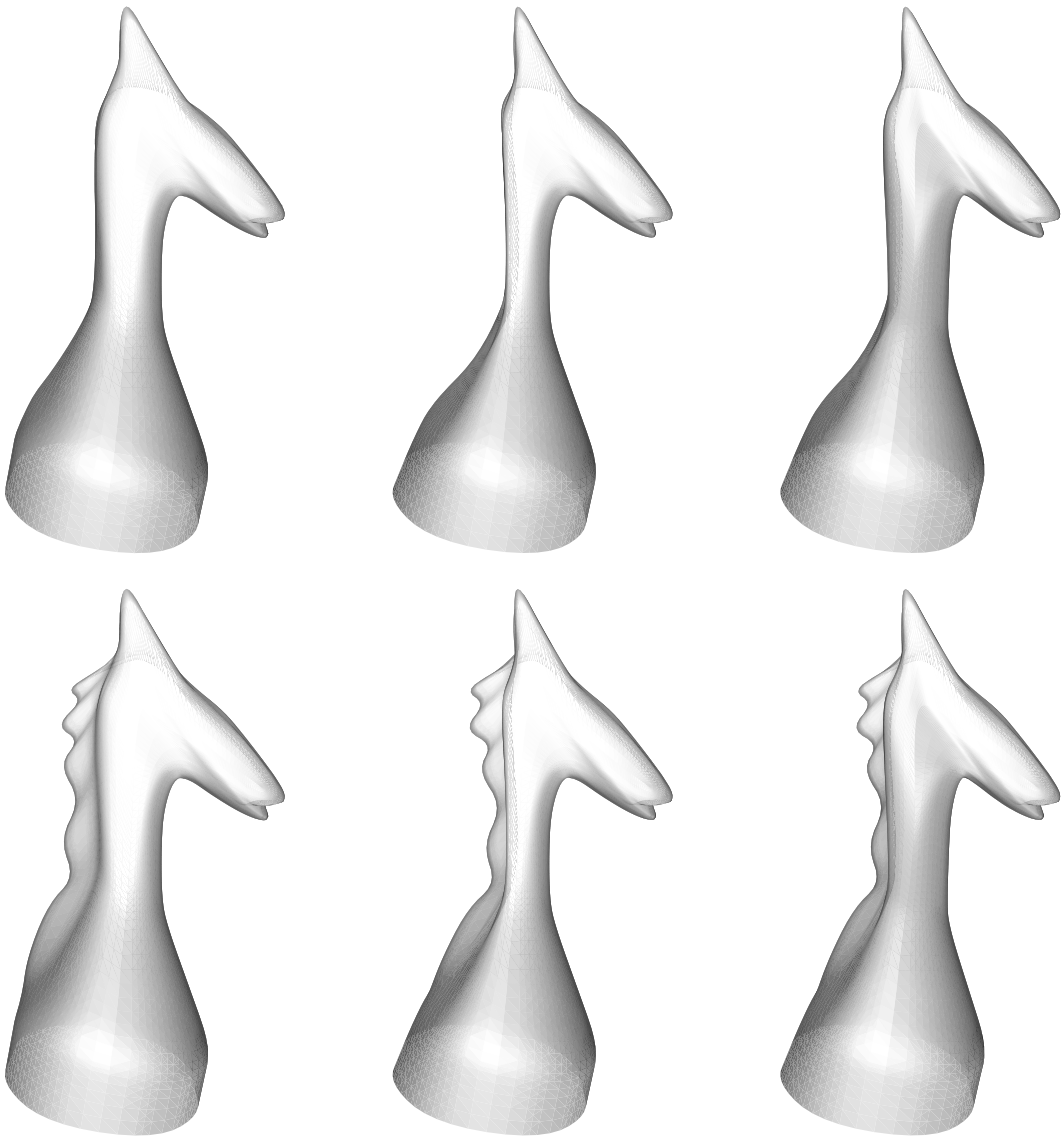


Fig. 3.13 – Déformation d'un cavalier, cas uniforme.
Les surfaces initiales (à gauche) sont déformées à l'encolure sans contrainte (au milieu) et avec conservation du volume (à droite).

larges déformations réalistes. Il serait judicieux de faire appel à un autre outil de manipulation, par exemple à l'aide d'un squelette. En écho aux remarques liminaires de ce manuscrit (section 1.4), rappelons que le cœur de ces travaux est l'étape de *correction* du volume, qui se veut justement indépendante de l'outil de manipulation (étape d'*édition*), qui n'est pour nous qu'un moyen de validation. La méthode de résolution est inchangée (pour le moins *a priori* adaptable) si la surface est éditée via un outil différent.

L'observation de cette figure 3.13 appelle par ailleurs un commentaire sur le modèle multi-résolution. L'avantage principal de ce type de modèle est de pourvoir déformer la surface à une certaine échelle sans modifier l'information aux autres échelles. Typiquement la crinière du cheval (détails haute fréquence, petite échelle) n'est pas dégradée par la déformation à une échelle plus globale. Elle "suit" le déplacement de la surface grossière qui la supporte, et cela sans que l'utilisateur ne s'en occupe : une fois définie l'échelle d'édition, les détails fins peuvent être conservés de façon transparente pour l'utilisateur.

Terminons cette étude sur une remarque concernant la continuité. Sur la surface en bas à droite on devine le raccord entre les deux patchs qui court de la base au bout du museau. Nous ne gérons en effet qu'une continuité C^0 entre les patchs, ce qui ressort au moment du rendu avec éclairage. Sans entrer dans les détails, la gestion d'un plus haut degré de continuité fixe des relations linéaires sur de plus nombreux points de part et d'autre de la couture. Il s'en suit des contraintes additionnelles dans les systèmes linéaires. Une telle extension est envisageable, et peut constituer d'intéressants travaux futurs. La principale difficulté est *a priori* la résolution du système linéaire : ce dernier sera plus grand et moins creux, donc moins facile à résoudre explicitement. Rappelons à ce sujet que la possibilité de résoudre le système explicitement, si elle n'est pas indispensable dans le cas des courbes, joue un rôle majeur dans l'efficacité pour les surfaces.

3.4 Comparaison

Les méthodes que nous avons présentées dans les sections 3.2 et 3.3 ont de nombreux points communs. Principalement elles atteignent l'objectif de déformer à différentes échelles des surfaces composites constituées de patchs B-spline produit tensoriel tout en conservant le volume englobé par cette surface. *A contrario*, un certain nombre d'aspects les différencient, tant structurellement que dans les possibilités applicatives. Nous démêlons ici les différences conceptuelles des différences accessoires. Les différences conceptuelles sont imposées par le modèle de surfaces, alors que les différences accessoires découlent des choix que nous avons faits pour résoudre le problème de déformation sous contrainte de volume.

Uniforme ou non uniforme ?

La principale différence conceptuelle entre la section 3.2 et la section 3.3 est l'utilisation des B-splines respectivement non uniformes et uniformes.

Ceci implique des possibilités de modélisation différentes : la méthode non uniforme est plus générale, en ce sens que la famille des surfaces représentables est plus grande. Notamment, bien que nous n'ayons pas abordé cet aspect, il est possible d'avoir des nœuds intérieurs multiples, ce qui permet par exemple des crêtes et des points anguleux. Les B-splines sont très utilisées dans le domaine de la conception géométrique assistée par ordinateur sous leur forme non uniforme pour cette liberté.

En contrepartie la représentation uniforme à l'avantage de sa simplicité. Là où la méthode non uniforme nécessite de travailler par insertion et suppression de nœuds, la méthode uniforme

utilise des filtres multirésolution pré-calculés et invariants par translation. Il s'en suit que les changements d'échelle sont plus simples et plus efficaces.

Multirésolution ou multi-échelle ?

La conséquence incontournable du choix entre uniforme et non uniforme est la différence entre les modèles à plusieurs échelles. Reprenons la nuance entre *multi-échelle* (cas non uniforme) et *multirésolution* (cas uniforme) introduite en section 2.5.2.

Une représentation multi-échelle est restreinte par rapport à une représentation multirésolution (comme présentée dans le chapitre introductif) car elle ne permet pas de *décomposer* la surface selon des coefficients d'approximation et des coefficients de détails. Elle permet de modifier à large échelle sans modifier les détails fins de la surface, mais pas l'inverse. Elle ne permet pas de travailler à plusieurs échelles à la fois. En reprenant la notion de boucle d'édition présentée en introduction, on s'aperçoit que le modèle multirésolution permet de travailler continuellement sur la surface décomposée, alors que le modèle multi-échelle impose de "remonter" à la surface fine pour chaque déformation, ce qui induit des calculs supplémentaires.

Nous remarquerons pourtant que ce que la méthode multirésolution présentée tire surtout profit d'une déformation large avec conservation des détails. Néanmoins, le simple fait qu'il existe un maillage de contrôle grossier (ce qui n'est pas le cas pour le multi-échelle) est précieux : celui-ci permet de visualiser une approximation de la surface. En outre, si l'on envisage une édition automatisée de la surface (par des contraintes extérieures à l'objet lui-même dans une scène complexe, ou via une déformation planifiée d'un squelette par exemple), ce maillage de contrôle est pratique : le déplacement de seulement quelques points de contrôle à un bas niveau de résolution permet de déplacer ou de déformer toute la surface.

En étant plus prospectif, si l'on envisage une déformation de la surface qui a besoin de coefficients à différentes échelles, le modèle multi-échelle atteint ces limites. Nous citerons à titre d'exemple la compression par ondelettes déjà discutée en section 2.5.2.

Point de contrôle ou point de la surface ?

Nous avons proposé des outils de manipulation différents selon les méthodes. Les méthodes multi-échelles sont basées sur le déplacement d'un point de la surface, et la méthode multirésolution est basée sur le déplacement d'un point de contrôle.

Pour en situer les conséquences, reportons-nous aux clés de lecture données dans l'introduction (section 1.4). Ces choix n'ont pas d'impact sur le calcul du volume en soi (dit *méthode de calcul*), mais influencent la façon d'appliquer la contrainte (dite *intégration*). En d'autres termes, le système linéaire à résoudre au final est légèrement différent selon le cas.

La manipulation via un point de la surface est exactement adaptable au cas uniforme. Elle a l'avantage ergonomique de spécifier précisément la position de la surface, ce qui peut être intéressant par exemple dans le cas où la déformation est causée par une collision avec d'autres objets.

La manipulation via un point de contrôle est théoriquement adaptable au cas non uniforme. Cela signifierait que l'utilisateur manipule directement les coefficients $\widetilde{\delta}_{\mathbf{j}}$ qui définissent la déformation dans l'espace d'approximation $\widetilde{V}$ (voir (3.4)). C'est peu intéressant dans la mesure où ces coefficients n'ont plus d'interprétation géométrique immédiate, alors que dans le cas uniforme ils représentent le déplacement d'un sommet du maillage de contrôle, ce qui est très intuitif.

Linéarisation de la contrainte.

Résumant les commentaires de la section 2.5.2, nous remarquons que la contrainte de volume a été (de façon interchangeable) linéarisée par isolement des axes dans un cas, et par un développement limité dans l'autre. La deuxième solution a l'inconvénient d'être approximante, ce qui oblige à faire plusieurs résolutions pour diminuer l'erreur. Par contre, elle est symétrique. Par ailleurs, nous conjecturons que les deux méthodes n'ont pas le même comportement dans certaines configurations géométriques pathologiques. Cela serait l'objet d'une analyse de stabilité numérique plus poussée.

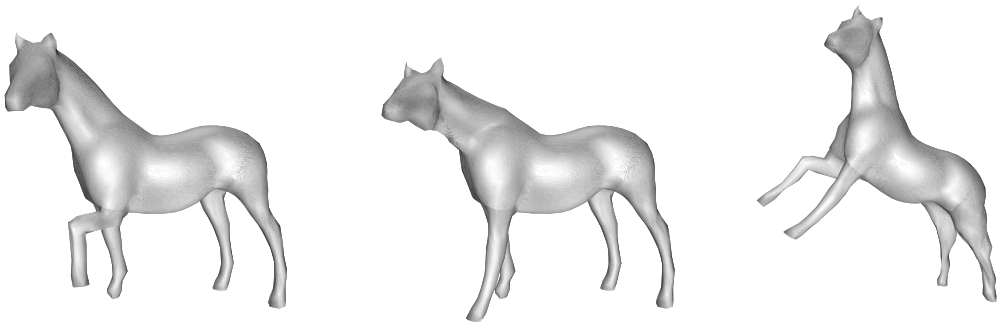
Contraintes linéaires additionnelles.

Quelle que soit la méthode, le principe général est de linéariser la contrainte de volume pour l'intégrer à faible coût algorithmique dans une minimisation. Les contraintes de position (point sur la surface ou point de contrôle) sont elles aussi linéaires. Il est donc possible d'intégrer relativement aisément de nouvelles contraintes telles que la spécification de tangentes, de normales ou de symétries. Ces possibilités sont traitées dans les travaux de Elber [Elb00, Elb01].

La conséquence directe est encore une fois que le système linéaire est moins creux, et sa taille ainsi que sa forme varient. Par conséquent, il n'est plus possible d'envisager une résolution explicite comme nous l'avons fait. Or si l'utilisation d'une méthode de résolution générique est lourde mais tolérable pour les courbes à aires constante, elle devient très lourde, voire rédhibitoire, pour les surfaces. À tout le moins les calculs en quasi temps-réel que nous exposons ici sont hors d'atteinte. Les tests que nous avons effectués montrent que les méthodes exposées dans ce chapitre nécessitent moins d'une seconde pour déformer une surface avec 5000 points de contrôle.

Chapitre 4

Volume et surfaces triangulaires



Nous présentons dans ce chapitre une méthode de déformation à volume constant de surfaces multirésolution de topologie arbitraire, paramétrées sur un maillage triangulaire. Le schéma multirésolution est basé sur la subdivision de Loop et s'appuie sur la technique du lifting scheme. Nous développons une méthode de calcul dans la base multirésolution du volume englobé par la surface, puis la déformation contrainte est définie à l'aide d'une minimisation sous contrainte. Nous apportons un soin particulier à l'étude des aspects algorithmiques des outils que nous construisons, afin de permettre une édition interactive de la surface.

4.1 Problématique

Les schémas de subdivision pour maillages triangulaires ont fait l'objet de recherches continues depuis une trentaine d'années. Cet engouement est vraisemblablement lié (i) au besoin de manipuler et représenter facilement des surfaces lisses, et (ii) au fait que les maillages triangulaires sont particulièrement maniables (donc répandus) pour les applications graphiques.

Afin d'allier l'efficacité algorithmique de la subdivision et la performance de l'analyse en ondelettes, des schémas multirésolution ont été construits *à partir* de schémas de subdivision. Des travaux marquants sont ceux de Lounsbery, DeRose et Warren [LDW97], qui ont montré qu'un schéma de subdivision définit une base de fonctions d'échelle. Formellement, la surface limite générée par le schéma de subdivision s'écrit dans cette base, avec pour coefficients les sommets du maillage avant subdivision. A condition que ce maillage soit semi-régulier, il est alors possible de définir des ondelettes, et par là un schéma multirésolution.

Parallèlement, Sweldens et ses collaborateurs [SS95, SS96, Swe97] ont développé des algorithmes efficaces de mise en œuvre des schémas multirésolution. Ils fonctionnent par accumulation de filtres simples, qui permettent de choisir les propriétés du schéma multirésolution résultant.

La combinaison de ces travaux nous fournit des modèles de surfaces multirésolution efficaces et modulables. À l'instar de Li *et al.* [LQS04], nous avons choisi de les utiliser pour construire un modèle multirésolution de surfaces de topologie quelconque, suivant nos propres critères : modéliser des surfaces lisses, et pouvoir les contrôler localement. Forts de ce choix, nous proposons dans ce chapitre une méthode de déformation à volume constant de ces surfaces.

Dans une première section (4.2), nous construisons une analyse multirésolution qui convient à nos critères. Dans un deuxième temps (section 4.3) nous présentons un calcul du volume, en insistant sur les difficultés algorithmiques. Dans la section 4.4 nous proposons une méthode de déformation de la surface intégrant la contrainte de volume. Enfin nos résultats sont commentés en section 4.5.

4.2 Schémas multirésolution sur des maillages triangulaires

La particularité (au sein des modèles en ondelettes) des surfaces que nous utilisons dans ce chapitre est d'être paramétrées sur un maillage triangulaire (par opposition à des pavés de $\mathbb{R}^2$ dans le chapitre 3). Ceci implique une construction particulière des espaces emboîtés V^j.

Dans la section 4.2.1 nous rappelons la notion topologique de maillages *semi-réguliers*. Nous construisons une suite de tels maillages qui nous servent à indexer les espaces emboîtés (cf. section 1.3). Nous appuyant sur des travaux existants [LDW97], nous rappelons qu'il existe des bases d'ondelettes et de fonctions d'échelles définissant une analyse multirésolution. Ces fonctions sont définies par un processus de subdivision.

Dans la section 4.2.2, nous montrons comment construire des filtres d'analyse et de synthèse répondant à nos besoins : régularité, support local et conservation de la moyenne. Nous faisons appel au formalisme des *lifting schemes* [SS96] pour modifier un schéma existant [LQS04, Ber04] basé sur la subdivision de Loop [Loo87]. Grâce à cela, nous aboutissons à un banc de filtres qui s'exécute en temps linéaire.

4.2.1 Existence de schémas multirésolution

Maillages semi-réguliers.

Soit un maillage triangulaire $\mathcal{M}^0$, variété de dimension 2. Considérons l'application sur $\mathcal{M}^0$ d'un schéma de subdivision primal, c'est à dire que nous construisons récursivement une séquence de maillages $\mathcal{M}^1, \mathcal{M}^2, \ldots, \mathcal{M}^n$ par quadrisection des faces. Ce processus est illustré Fig. 4.1 : à chaque étape $\mathcal{M}^j \rightarrow \mathcal{M}^{j+1}$ un sommet est inséré au milieu de chaque arête et chaque face est divisée en 4. Le maillage $\mathcal{M}^n$ ainsi construit est semi-régulier (comme tous les intermédiaires $\mathcal{M}^j$), ce qui signifie qu'il n'a qu'un petit nombre de sommets irréguliers (au plus ceux du maillage $\mathcal{M}^0$, tous les autres étant de valence 6) et que, étant issu d'un processus de subdivision, il est possible de lui appliquer le processus inverse par fusion des faces 4 par 4.

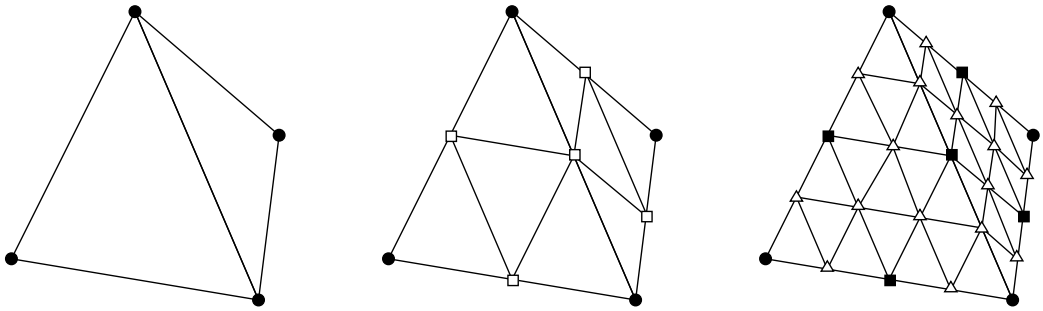


Fig. 4.1 – Subdivision d'un tétraèdre.
De gauche à droite : $\mathcal{M}^0$, $\mathcal{M}^1$ et $\mathcal{M}^2$. Les nouveaux sommets ($\mathcal{W}^0 = \{\square\}$ puis $\mathcal{W}^1 = \{\triangle\}$) sont en blanc, les anciens ($\mathcal{V}^0 = \{\bullet\}$ puis $\mathcal{V}^1 = \{\bullet, \blacksquare\}$) sont en noir.

Ce processus réversible nous fournit la base d'un schéma multirésolution, dans lequel les sommets des maillages $\mathcal{M}^j$ *sont les indices* des coefficients d'échelles et d'ondelettes. Quant à lui, le maillage $\mathcal{M}^0$ sera l'espace de paramétrisation des fonctions de base et de la surface limite lisse. Pour appréhender ce qui va suivre il faut bien distinguer trois objets :

- Les maillages $\mathcal{M}^j$ sont des polytopes, sur lesquels nous appliquons un schéma de subdivision au sens *topologique*. C'est-à-dire que nous définissons des faces et des sommets avec des relations d'adjacence.
- Les maillages *géométriques*, appelés maillages de contrôle, seront définis dans l'espace 3D par des *points de contrôle* qui associent une position dans $\mathbb{R}^3$ à un sommet de $\mathcal{M}^j$.
- Les surfaces lisses s'exprimeront dans les bases d'ondelettes et de fonctions d'échelles par leurs *coefficients*. Il peut s'agir de coefficients d'échelles (*i.e.* points de contrôle) ou de coefficients d'ondelettes.

Notons $\mathcal{V}^j$ l'ensemble des sommets du maillage $\mathcal{M}^j$ pour $j \in \{0, \ldots, n\}$. Chaque ensemble $\mathcal{V}^j$ est l'union de $\mathcal{V}^{j-1}$, et de l'ensemble des sommets insérés par l'étape de subdivision $\mathcal{M}^{j-1} \rightarrow \mathcal{M}^j$, que nous noterons $\mathcal{W}^{j-1}$. Nous appellerons *sommets d'angle* les sommets de $\mathcal{V}^{j-1}$, et *sommets d'arête* les sommets de $\mathcal{W}^{j-1}$, car ces derniers sont insérés au milieu d'une arête de $\mathcal{M}^{j-1}$. Pour n'importe quel *niveau de résolution* $e \in \{0, \ldots, n\}$, l'ensemble $\mathcal{V}^n$ de tous les sommets de $\mathcal{M}^n$ admet une partition :

$$\mathcal{V}^n = \mathcal{V}^e \cup \mathcal{W}^e \cup \mathcal{W}^{e+1} \cup \ldots \cup \mathcal{W}^{n-1} \ . \tag{4.1}$$

Existence d'espaces emboîtés basés sur la subdivision.

À partir des maillages ci-dessus, il est possible de construire [LDW97] une suite d'espaces emboîtés $V^0 \subset V^1 \subset \cdots \subset V^n$, tels que nous les avons définis dans la section introductive 1.3, et qui sont le fondement d'une analyse multirésolution. Cela revient à définir les bases de fonctions d'échelles pour V^j. À chaque niveau j ces dernières se notent

$$\begin{array}{rccc} \varphi_i^j & : & \mathcal{M}^0 & \longrightarrow & \mathbb{R} \\ & & t & \longmapsto & \varphi_i^j(t) \end{array}$$

pour $i \in \mathcal{V}^j$. Elles sont paramétrées sur le maillage $\mathcal{M}^0$, et indexées sur les sommets $\mathcal{V}^j$.

Pour construire ces fonctions, Lounsbery, DeRose et Warren [LDW97] proposent l'utilisation d'un schéma de subdivision. Supposons que l'on dispose d'un schéma de subdivision (au sens *géométrique*), primal et convergent. Deux schémas classiques sont le schéma interpolant Butterfly [DLG90], et le schéma approximant de Loop [Loo87] (voir FIG. 4.2). En affectant la valeur 1 (dans $\mathbb{R}$) au sommet $i \in \mathcal{V}^j$ de $\mathcal{M}^j$ et 0 à tous les autres, le schéma de subdivision fait converger l'image du maillage vers une fonction à valeurs réelles, paramétrée sur $\mathcal{M}^0$, et cette limite est φ_i^j. Pour la démonstration nous invitons le lecteur à se référer aux travaux de Lounsbery *et al* [LDW97].

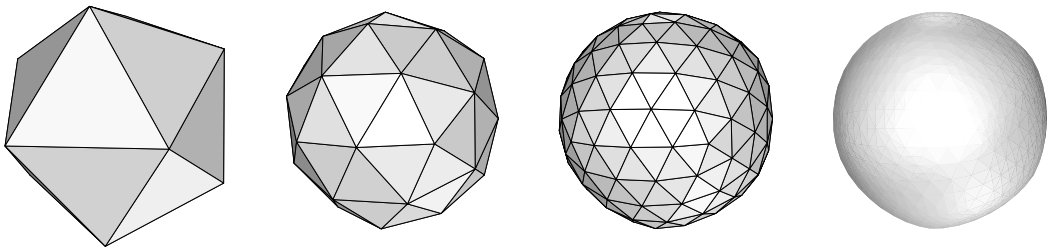


FIG. 4.2 – Subdivision de Loop.
De gauche à droite : maillage initial (icosaèdre), après une étape de subdivision, après deux étapes. La surface limite (à droite) est de classe C^2, excepté aux sommets extraordinaires.

Nous disposons alors de bases de fonctions pour V^j, $j \in \{0, \ldots, n\}$, notées sous forme de vecteurs colonnes $\varphi^j = (\varphi_i^j)_{i \in \mathcal{V}^j}$, et nous pouvons définir une surface $S(t)$ de l'espace V^n comme

$$S(t) = \sum_{i \in \mathcal{V}^n} \mathbf{c}_i^n \, \varphi_i^n(t) \ , \quad \text{pour } t \in \mathcal{M}^0 \ , \tag{4.2}$$

où les $\mathbf{c}_i \in \mathbb{R}^3$ sont les coefficients d'échelles au niveau n, aussi sommets du polyèdre de contrôle de S. Les filtres de subdivision se formalisent sous la forme de matrices de subdivision P^j, $j \in \{1, \ldots, n\}$ et les fonctions d'échelles vérifient les relations dites "à deux échelles" suivantes :

$$\varphi^{j-1} = (P^j)^T \varphi^j \quad \text{pour } j \in \{1, \ldots, n\}.$$

Existence des ondelettes.

Afin de finaliser le schéma multirésolution, il nous faut définir les espaces de détails W^j, complémentaires de V^j dans V^{j+1} (voir section 1.3). Nous avons besoin de filtres de correction,

c'est-à-dire de matrices de correction Q^j, $j \in \{1,\dots,n\}$, à l'aide desquelles $(n-1)$ familles d'ondelettes $(\psi_i^j)_{i\in\mathcal{W}^j}$, $j \in \{0,\dots,n-1\}$ sont définies par

$$\psi^{j-1} = (Q^j)^T \varphi^j \quad \text{pour } j \in \{1,\dots,n\}.$$

Ces familles forment des bases pour des espaces W^j. Lounsbery *et al* proposent de construire ces filtres de correction grâce à des critères locaux d'orthogonalité entre les fonctions de base [LDW97]. Nous présentons et justifions dans la prochaine section l'utilisation d'une autre méthode plus appropriée à nos objectifs.

De fait, parallèlement à la partition des indices (4.1), nous avons une décomposition de l'espace $V^n = V^e \oplus W^e \oplus W^{e+1} \oplus \cdots \oplus W^{n-1}$ pour un niveau d'analyse $e \in \{0,\dots,n\}$ donné. Cette décomposition s'accompagne d'une base multirésolution $\{\varphi^e, \psi^e, \dots, \psi^{n-1}\}$ constituée de fonctions d'échelles au niveau e et d'ondelettes aux niveaux $e, e+1, \dots, n-1$. Dans cette nouvelle base une surface $S \in V^n$ s'exprime par l'équation (4.3) à l'aide de coefficients d'échelle $\mathbf{c}_i^e$, $i \in \mathcal{V}^e$, et de coefficients d'ondelettes $\mathbf{d}_i^j$, $i \in \mathcal{W}^j$, pour chaque niveau $j = e, e+1, \dots, n-1$:

$$S(t) = \sum_{i\in\mathcal{V}^e} \mathbf{c}_i^e\, \varphi_i^e(t) + \sum_{j=e}^{n-1} \sum_{i\in\mathcal{W}^j} \mathbf{d}_i^j\, \psi_i^j(t). \tag{4.3}$$

Filtres d'analyse.

Pour effectuer les changements de bases nécessaires à l'expression multirésolution de S (Eq. (4.3)), il nous faut définir pour chaque échelle $j \in \{1,\dots,n\}$ des filtres d'analyse A^j et B^j par

$$\begin{bmatrix} A^j \\ B^j \end{bmatrix} = \begin{bmatrix} P^j & Q^j \end{bmatrix}^{-1}.$$

Nous verrons en pratique (section 4.2.2) qu'il n'est pas nécessaire de construire ces filtres par inversion matricielle.

Nous disposons maintenant de tous les filtres pour construire un banc de filtres (voir Fig. 1.1). La figure 4.3 illustre ce processus qui permet de décomposer la surface (exprimée en base fine) par des étapes d'analyse vérifiant

$$\mathbf{c}^{j-1} = A^j \mathbf{c}^j \quad \text{et} \quad \mathbf{d}^{j-1} = B^j \mathbf{c}^j.$$

Les matrices A^j sont communément nommées filtres *passe bas*, et les matrices B^j filtres *passe haut*. Réciproquement la surface (exprimée en base multirésolution) est reconstruite par des étapes de synthèse vérifiant

$$\mathbf{c}^j = P^j \mathbf{c}^{j-1} + Q^j \mathbf{d}^{j-1}.$$

Les vecteurs colonnes $\mathbf{c}^j$ désignent $(\mathbf{c}_i^j)_i$ et $\mathbf{d}^j = (\mathbf{d}_i^j)_i$. Les maillages de contrôle, notés $\mathbf{c}^j(\mathcal{M}^j)$, sont l'image dans $\mathbb{R}^3$ des maillages topologiques $\mathcal{M}^j$, où chaque point $\mathbf{c}_i^j$ est associé au sommet $i \in \mathcal{V}^j$.

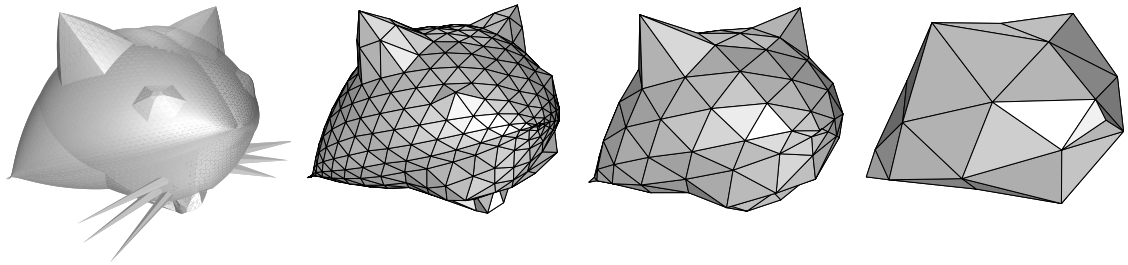


FIG. 4.3 – Analyse multirésolution d'un maillage semi-régulier.
Le maillage fin a 16384 faces ($n = 6$). Les maillages de contrôle sont représentés pour $e \in \{6, 4, 3, 2\}$ de gauche à droite : $\mathbf{c}^6(\mathcal{M}^6)$, $\mathbf{c}^4(\mathcal{M}^4)$, $\mathbf{c}^3(\mathcal{M}^3)$ et $\mathbf{c}^2(\mathcal{M}^2)$.

4.2.2 Construction d'un schéma multirésolution par lifting scheme

Nous savons désormais qu'il existe des schémas multirésolution pour des surfaces construites à partir d'un maillage semi-régulier. Mais dans la section précédente, nous n'avons pas construit certains filtres. L'écriture sous forme matricielle n'est en effet pas pratique pour l'implémentation, notamment pour des raisons de place mémoire : la matrice $[P^n\, Q^n]$ par exemple est de taille $card(\mathcal{V}^n) \times card(\mathcal{V}^n)$, c'est à dire le carré du nombre de sommets du maillage. De plus l'application de ces filtres nécessite généralement l'allocation de mémoire auxiliaire.

Par ailleurs, le choix des filtres dépend des applications, et il n'est pas aisé d'en construire qui ont toutes les propriétés dont on a besoin. Classiquement en traitement du signal, il est souhaitable que les ondelettes aient un certain nombre de moments nuls, que la décomposition ait de bonnes propriétés d'approximation L_2, et que les bases vérifient des conditions d'orthogonalité. En ce qui nous concerne, les critères sont autres :

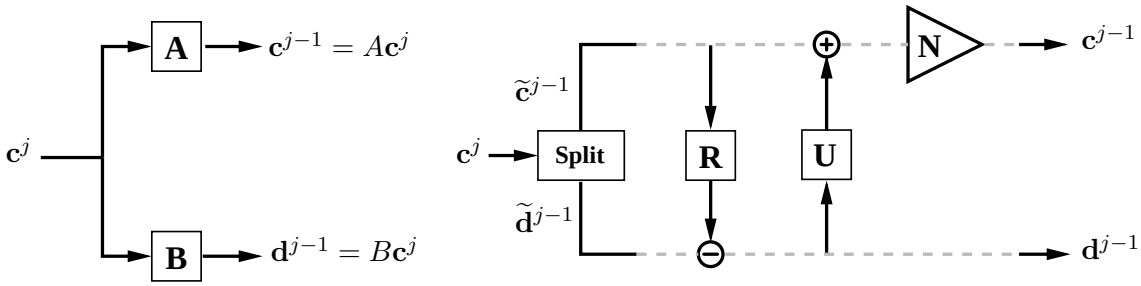
Régularité : il est intéressant que les fonctions d'échelles soient régulières pour représenter des surfaces lisses.

Support compact : il est très important que les fonctions de base aient un support compact. D'une part les coefficients ont ainsi une influence locale, ce qui permet de localiser une déformation. D'autre part les filtres sont creux, donc chaque étape du banc de filtres se fait en temps linéaire par rapport au nombre de sommets.

Conservation de la moyenne : pour des raisons pratiques lors de l'édition, il est avantageux que la moyenne des coefficients d'échelle $\mathbf{c}_i^j$ ne dépende pas de j. Cela évite que le maillage de contrôle, à l'aide duquel nous allons manipuler la surface, ne s'écarte trop de cette dernière.

Pour surmonter les obstacles que nous venons de citer (occupation mémoire et construction d'un schéma approprié à nos critères), nous faisons appel à la technique de *lifting scheme* développée par Sweldens [Swe97]. Il s'agit d'une méthode de construction *et* d'application des filtres complexes A, B, P et Q, qui procède par applications successives de filtres simples. C'est une approche pragmatique et algorithmique. Prenons une étape de décomposition

$$\begin{pmatrix} \mathbf{c}^{j-1} \\ \mathbf{d}^{j-1} \end{pmatrix} = \begin{bmatrix} A \\ B \end{bmatrix} \begin{pmatrix} \\ \mathbf{c}^j \\ \\ \end{pmatrix} .$$



(a) Approche directe (b) Lifting scheme

Fig. 4.4 – Comparaison entre l'approche directe et le lifting scheme.

L'approche directe consiste à appliquer les filtres A et B, aussi complexes soient-ils, sous forme matricielle ou à l'aide de masques de convolution. L'idée du lifting scheme, formalisée figure 4.4, est de commencer par répartir les coefficients $\mathbf{c}_i^j$ en deux parties $\widetilde{\mathbf{c}}^{j-1} = \left(\mathbf{c}_i^j\right)_{i\in\mathcal{V}^{j-1}}$ et $\widetilde{\mathbf{d}}^{j-1} = \left(\mathbf{c}_i^j\right)_{i\in\mathcal{W}^{j-1}}$, selon que $i \in \mathcal{V}^{j-1}$ ou que $i \in \mathcal{W}^{j-1}$. Cette séparation triviale est en soi déjà une analyse multirésolution, dite *lazy-wavelets* car elle ne fait rien d'autre que de sous-échantillonner le signal. Ensuite cette décomposition est améliorée par différentes opérations de filtrage, appelées *phases* :

Prédiction : partant des coefficients d'échelle $\widetilde{\mathbf{c}}^{j-1}$, une prédiction $R\widetilde{\mathbf{c}}^{j-1}$ de $\widetilde{\mathbf{d}}^{j-1}$ est faite, et les coefficients d'ondelettes sont modifiés de manière à coder l'écart à cette prédiction :

$$\widetilde{\mathbf{d}}^{j-1} = \widetilde{\mathbf{d}}^{j-1} - R\widetilde{\mathbf{c}}^{j-1} \ .$$

La prédiction peut se faire dualement sur les coefficients d'échelles :

$$\widetilde{\mathbf{c}}^{j-1} = \widetilde{\mathbf{c}}^{j-1} - R\widetilde{\mathbf{d}}^{j-1} \ .$$

Mise à jour : une telle phase consiste à mettre à jour $\widetilde{\mathbf{c}}^{j-1}$ par addition d'une combinaison linéaire de coefficients de détails :

$$\widetilde{\mathbf{c}}^{j-1} = \widetilde{\mathbf{c}}^{j-1} + U\widetilde{\mathbf{d}}^{j-1} \ .$$

Il est aussi possible de faire une mise à jour duale $\widetilde{\mathbf{d}}^{j-1} = \widetilde{\mathbf{d}}^{j-1} + U\widetilde{\mathbf{c}}^{j-1}$.

Mise à l'échelle : cette phase consiste à multiplier les coefficients $\widetilde{\mathbf{c}}_i^{j-1}$ (ou $\widetilde{\mathbf{d}}_i^{j-1}$) par une valeur non nulle (qui peut dépendre de i) :

$$\widetilde{\mathbf{c}}^{j-1} = N\widetilde{\mathbf{c}}^{j-1} \quad \text{ou} \quad \widetilde{\mathbf{d}}^{j-1} = N\widetilde{\mathbf{d}}^{j-1} \ ,$$

où N est une matrice diagonale (à coefficients diagonaux non nuls). N est parfois appelé "gain" [Swe96, Swe97].

Après plusieurs de ces opérations, les coefficients finaux $\mathbf{c}^{j-1} = \widetilde{\mathbf{c}}^{j-1}$ et $\mathbf{d}^{j-1} = \widetilde{\mathbf{d}}^{j-1}$ sont obtenus.

Cette formalisation a plusieurs intérêts :

Isolement des degrés de liberté : l'étape de répartition (*split* $\mathbf{c}^j \rightarrow (\widetilde{\mathbf{c}}^{j-1}, \widetilde{\mathbf{d}}^{j-1})$) conserve toute l'information sans redondance : c'est un changement de base. Les phases de prédiction modifient les coefficients de détail $\widetilde{\mathbf{d}}^{j-1}$ à l'aide d'une combinaison linéaire des

coefficients d'échelles $\widetilde{\mathbf{c}}^{j-1}$ (ou réciproquement), qui sont deux ensembles distincts. Quel que soit R, il s'agit donc aussi d'un changement de base. Il en va de même pour les phases de mise à jour. Les mises à l'échelle sont de toute évidence des changements de base, du moment que les coefficients diagonaux sont non nuls.
En d'autres termes, les choix de R, U et N sont libres de toute contrainte (*e.g.* inversibilité) : quels qu'ils soient, la matrice d'analyse $\begin{bmatrix} A \\ B \end{bmatrix}$ résultante est inversible. On peut ainsi déterminer R, U et N dans le seul souci de satisfaire nos objectifs de régularité et de support compact.

Efficacité mémoire : puisque, au cours d'une phase, les coefficients modifiés sont distincts de ceux qui entrent en jeu dans les combinaisons linéaires, il n'est pas nécessaire d'allouer de mémoire supplémentaire pour appliquer les filtres.

Facilité algorithmique : chaque phase peut être locale (matrices R et U très creuses), ce qui évite au programmeur le codage fastidieux de filtres larges.

Réversibilité : pour appliquer les filtres de synthèse il suffit de renverser le schéma (voir FIG. 4.8 qui présente un schéma basé sur la subdivision de Loop, que nous construisons ultérieurement). Les différentes phases sont appliquées en ordre inverse, et en inversant $+ \leftrightarrow -$ et $\times \leftrightarrow \div$. La séparation (*split*) est remplacée par une fusion (*merge*).

Modularité : il est possible de mettre bout-à-bout (et sans contrainte sur l'ordre) plusieurs phases de son choix pour assurer différentes propriétés au schéma multirésolution final.

Subdivision de Loop sous forme de lifting scheme.

Suivant les conseils de Sweldens et Schröder [SS96], une démarche aisée pour construire un lifting scheme est de définir les filtres de reconstruction, desquels découleront les filtres d'analyse. Pour cela il faut d'abord choisir un schéma de subdivision et le formuler en terme de lifting. C'est l'objet de cette section.

Pour effectuer ce choix, rappelons que le schéma doit être primal. Prenons ensuite en compte deux de nos critères initiaux :

- La surface limite doit être lisse, ce qui nous invite à choisir le schéma de Loop [Loo87], qui converge vers une surface box-spline de classe C^2, excepté aux sommets extraordinaires. Il est reconnu que le schéma Butterfly [DLG90], son principal concurrent, a de moins bons résultats visuels, notamment pour la visualisation avec éclairage. L'avantage de ce dernier est d'être interpolant, ce qui n'est pas très intéressant pour nous, d'autant que le maillage de contrôle a tendance à se trouver à l'intérieur de la surface limite (dans les parties convexes), ce qui est peu ergonomique.
- Les filtres doivent être aussi creux que possible, afin que l'analyse et la synthèse soient peu coûteux. De ce point de vue les deux schémas sont sensiblement équivalents.

Par conséquent nous avons choisi le schéma de subdivision de Loop. Nous référant aux travaux de Li *et al.* [LQS04], ce schéma peut s'exprimer par le lifting scheme représenté figure 4.5. En entrée seuls les points d'angle $\mathbf{c}_i^{j-1}$ existent, les points d'arête $\mathbf{c}_i^j$, $i \in \mathcal{W}^{j-1}$, sont créés par l'application de R_i. Le lifting est constitué de deux phases de prédiction (R_i primale et R_p duale), et d'une phase de mise à l'échelle (N). Ces trois filtres ne sont pas construits sous forme de matrices : elles sont codés par des masques locaux qui s'appliquent à chacun des sommets. Les masques de R_i et R_p sont donnés FIG. 4.6 (a) et (b) pour les sommets intérieurs au maillage.

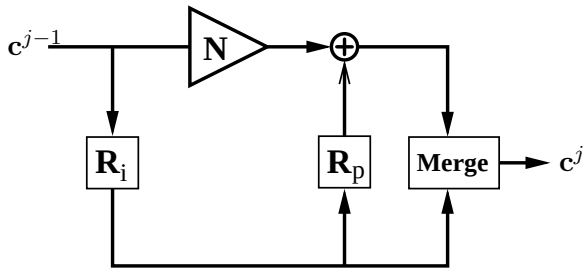


Fig. 4.5 – Subdivision de Loop sous forme de lifting scheme.
Ce schéma représente une étape de subdivision $\mathbf{c}^{j-1}(\mathcal{M}^{j-1}) \rightarrow \mathbf{c}^j(\mathcal{M}^j)$. Les points d'angle ($i \in \mathcal{V}^{j-1}$) passent par la branche du haut, et les points d'arête sont créés par R_i dans la branche du bas.

Le masque de R_p pour un sommet doit être multiplié par $\frac{5}{8}\beta$, où

$$\beta = \frac{1}{k}\left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4}\cos\left(\frac{2\pi}{k}\right)\right)^2\right) \quad \text{avec } k \text{ la valence du sommet.} \tag{4.4}$$

L'application du *filtre* R_p (Fig. 4.5) est l'application du *masque de* R_p (Fig. 4.6(b)) sur tous les sommets d'angle $f \in \mathcal{V}^{j-1}$:

pour $f \in \mathcal{V}^{j-1}$ **faire**
  β est défini par Eq. (4.4), où k est la valence de f
  pour $g \in \mathcal{W}^{j-1}$ voisin de f **faire**
    $\mathbf{c}_f^{j-1} = \mathbf{c}_f^{j-1} + \frac{5}{8}\beta\mathbf{c}_g^j$
  fin pour
fin pour

Cela revient à dire que la ligne f de la matrice R_p contient $R_p(f,g) = \frac{5}{8}\beta_f$ pour tous les indices de colonne $g \in \mathcal{W}^{j-1}$ tels que g est un voisin de f.

De la même façon, l'application du *filtre* R_i (Fig. 4.5) est l'application du *masque de* R_i (Fig. 4.6(a)) sur tous les sommets d'arête $g \in \mathcal{W}^{j-1}$. Remarquons que le masque du filtre R_i modifie le coefficient $\mathbf{c}_g^j$ lié à un sommet d'arête $g \in \mathcal{W}^{j-1}$, en lui ajoutant une combinaison linéaire de points $\mathbf{c}_f^{j-1}$ liés à des sommets d'angle $f \in \mathcal{V}^{j-1}$. Or $\mathcal{W}^{j-1}$ et $\mathcal{V}^{j-1}$ sont des ensembles distincts. C'est exactement pour cette raison que le filtre R_i s'applique "en place", c'est-à-dire sans allocation mémoire supplémentaire. Il en va ainsi pour tous les filtres du lifting.

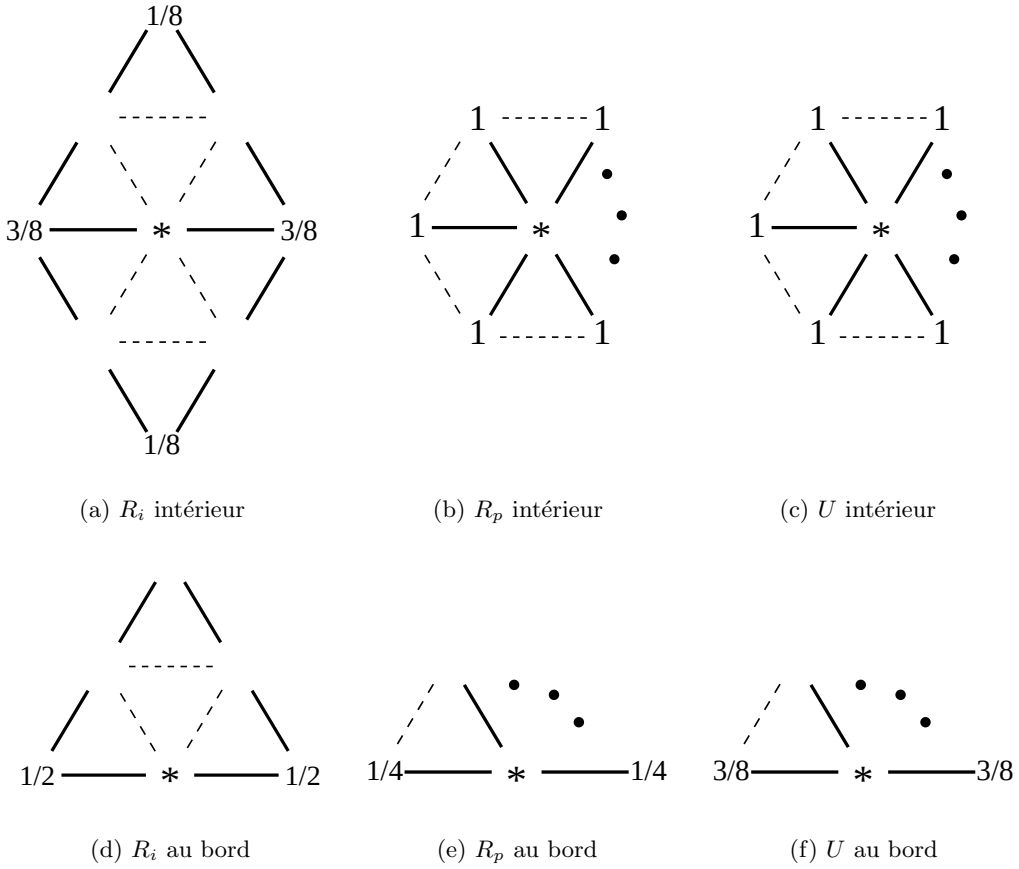
Les masques sont adaptés aux bords (Fig. 4.6 (d) et (e)) de telle sorte que les courbes de bord ne dépendent que des points du bord. Cette remarque est importante pour nous, car, lors des déformations, nous devrons conserver les courbes de bord. Il suffira donc de fixer les points de bord du maillage.

La phase N de mise à l'échelle multiplie chaque point d'angle $\mathbf{c}_i^{j-1}$, $i \in \mathcal{V}^{j-1}$, par un facteur

$$N_i = 1 - \frac{8}{5}k_i\beta_i = \frac{8}{5}\left(\frac{3}{8} + \frac{1}{4}\cos\left(\frac{2\pi}{k}\right)\right)^2 \quad \text{pour un sommet intérieur,}$$

$$N_i = \frac{1}{2} \quad \text{pour un sommet de bord.}$$

L'application des filtres du lifting par les masques de la figure 4.6 est strictement équivalente à l'application des masques définis par Loop [Loo87], que nous rappelons figure 4.7. La différence



(a) R_i intérieur (b) R_p intérieur (c) U intérieur

(d) R_i au bord (e) R_p au bord (f) U au bord

FIG. 4.6 – Masques de lifting scheme pour le schéma de Loop.
Ligne du haut : pour des sommets intérieurs. Ligne du bas : pour des sommets au bord du maillage. Le sommet $*$ est le sommet autour duquel le masque s'applique. Le masque pour R_p intérieur doit être normalisé par $\frac{5}{8}\beta$, et celui pour U intérieur par $\frac{5+16\beta}{10+5k}$, avec $\beta = \frac{1}{k}(\frac{5}{8} - (\frac{3}{8} + \frac{1}{4}\cos(\frac{2\pi}{k}))^2)$, où k est la valence du sommet $*$. Les arêtes en trait plein sont des demi-arêtes de $\mathcal{M}^{j-1}$ dans le processus de subdivision $\mathcal{M}^{j-1} \rightarrow \mathcal{M}^j$, alors que celles en pointillés sont des arêtes de $\mathcal{M}^j$ à l'intérieur des faces de $\mathcal{M}^{j-1}$.

est que les masques définis par Loop ne peuvent pas s'appliquer "en place", car ils pondèrent des points d'angle pour définir d'autres points d'angle (Fig. 4.7 à droite).

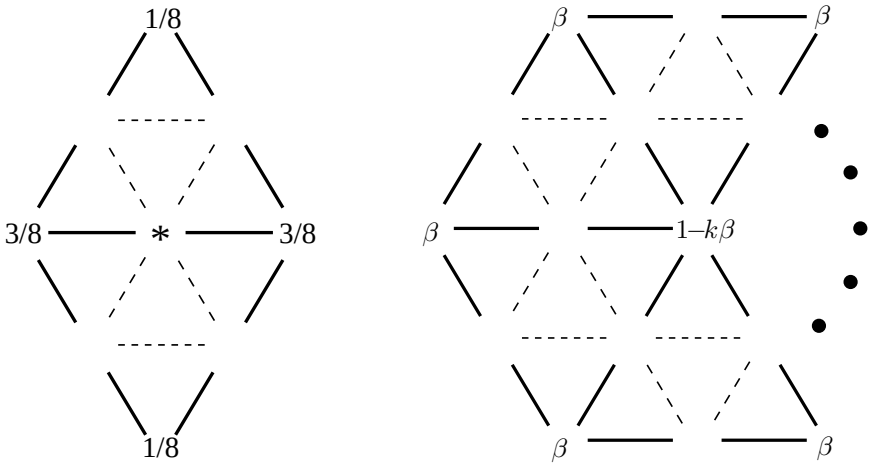


Fig. 4.7 – Masques de subdivision de Loop.
À gauche : filtre pour un sommet d'arête $i \in \mathcal{W}^{j-1}$.
À droite : filtre pour un sommet d'angle $i \in \mathcal{V}^{j-1}$.

Schéma multirésolution basé sur la subdivision de Loop, sans phase de mise à jour.

R_i, R_p et N étant définis, il existe un schéma multirésolution tel que le filtre de subdivision est la subdivision de Loop. La formalisation lifting scheme est donnée dans la figure 4.8 sans la phase U. Ce schéma a pour principales caractéristiques :

i. Dans l'écriture matricielle, les lignes du filtre de subdivision $\widetilde{P}$ contiennent les coefficients des masques de subdivision de Loop.

ii. Le filtre de correction $\widetilde{Q}$ associé vérifie :

$$\begin{bmatrix} & \\ \widetilde{P} & \widetilde{Q} \\ & \end{bmatrix} = \begin{bmatrix} I & R_p \\ 0 & I \end{bmatrix} \begin{bmatrix} N & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} I & 0 \\ R_i & I \end{bmatrix}$$

Cette écriture apporte une interprétation intuitive du fait que chaque phase du lifting opère exclusivement sur les coefficients de détails ou sur les coefficients d'échelles. En corollaire, la matrice de synthèse est inversible car la matrice de chaque phase est trivialement inversible.

iii. Les fonctions d'échelles φ^j sont les fonctions Box-splines servant à exprimer la surface limite du schéma de subdivision.

iv. Pour terminer la construction du schéma multirésolution, il nous faut construire les filtres d'analyse. Un avantage du lifting scheme est que l'inversion du processus de synthèse se fait aisément le "retournant" le schéma (voir figure 4.8 sans la phase U). Les différentes phases sont appliquées en ordre inverse, les additions et les soustractions sont inversées (phases de prédiction et de mise à jour), les multiplications et les divisions sont aussi inversées (phases

de mise à l'échelle). Enfin la séparation (*split*) est remplacée par une fusion (*merge*). Tout comme la synthèse, l'analyse peut s'interpréter matriciellement comme une écriture des filtres de synthèse $\widetilde{A}$ et $\widetilde{B}$ par un produit de matrices :

$$\begin{bmatrix} \widetilde{A} \\ \widetilde{B} \end{bmatrix} = \begin{bmatrix} I & 0 \\ -R_i & I \end{bmatrix} \begin{bmatrix} N^{-1} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} I & -R_p \\ 0 & I \end{bmatrix} .$$

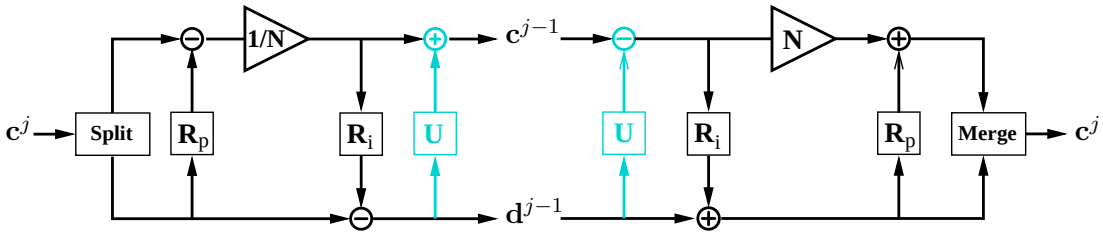


FIG. 4.8 – Lifting scheme basé sur le schéma de Loop.
L'analyse (à gauche) décompose $\mathbf{c}^j$ en $(\mathbf{c}^{j-1}, \mathbf{d}^{j-1})$ et la synthèse (à droite) reconstruit $\mathbf{c}^j$.

À ce point, nous disposons d'un schéma multirésolution qui vérifie deux des trois critères que nous nous sommes fixés : il modélise des surfaces lisses, et il permet un contrôle local. Pour satisfaire au troisième critère (conservation de la moyenne), nous proposons d'ajouter une phase de mise à jour (phase U FIG. 4.8) qui s'insère avant la prédiction primale dans le lifting de synthèse, après dans le lifting d'analyse. L'écriture matricielle de ce nouveau schéma est

$$\begin{bmatrix} P & Q \end{bmatrix} = \begin{bmatrix} \widetilde{P} & \widetilde{Q} \end{bmatrix} \begin{bmatrix} I & -U \\ 0 & I \end{bmatrix} \quad \text{et} \quad \begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} I & U \\ 0 & I \end{bmatrix} \begin{bmatrix} \widetilde{A} \\ \widetilde{B} \end{bmatrix} .$$

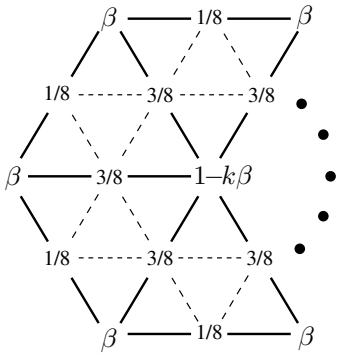
Phase de mise à jour pour améliorer les propriétés du schéma multirésolution.

Les approches existantes [LQS04, Ber04] proposent d'améliorer le schéma en suivant la même stratégie que Lounsbery [LDW97], qui consiste à orthogonaliser partiellement les ondelettes. Cette stratégie est intéressante si l'on cherche des propriétés d'approximation L^2, par exemple dans le but de compresser des maillages. Notre intérêt est plutôt que les ondelettes soient d'intégrale nulle. D'une part, ceci implique que la moyenne des sommets du maillage de contrôle $\mathbf{c}^j(\mathcal{M}^j)$ qui s'écrit

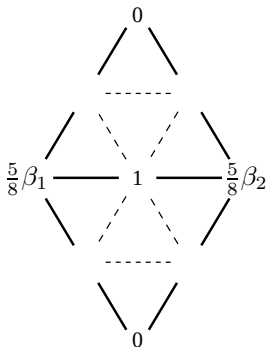
$$\frac{1}{card(\mathcal{V}^j)} \sum_{i \in \mathcal{V}^j} \mathbf{c}_i^j ,$$

est indépendante du niveau de résolution j. D'autre part, nous évitons ainsi que le maillage de contrôle rétrécisse (ou au contraire grandisse) à chaque étape d'analyse. Nous nous proposons ici de définir une phase de mise à jour qui atteint cet objectif.

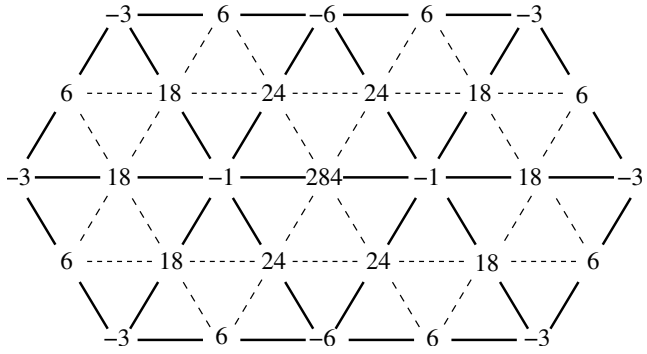
Pour définir cette phase, il nous faut introduire deux nouveaux types de masques, que nous nommerons masques *basse fréquence* et masques *haute fréquence*. Ils se rapportent respectivement aux sommets d'angle et aux sommets d'arête. Dans la littérature, ils sont habituellement



(a) masque basse fréquence



(b) masque haute fréquence sans mise à jour



(c) masque haute fréquence avec mise à jour (sommets réguliers)

Fig. 4.9 – Masques haute et basse fréquence basés sur le schéma de Loop.
Masques locaux définissant les filtres P et Q du schéma multirésolution fondé sur la subdivision de Loop.
Les masques (a) et (b) ne prennent pas en compte la phase de mise à jour. Le masque (c) prend en compte la mise à jour et doit être normalisé par $\frac{1}{320}$. Il correspond aux régions régulières du maillage. Le masque (a) correspond aux sommets d'angle de valence quelconque, avec $\beta = \frac{1}{k}(\frac{5}{8} - (\frac{3}{8} + \frac{1}{4}\cos(\frac{2\pi}{k}))^2)$, où k est la valence du sommet central. Le masque (b) correspond aux sommets d'arête, quelles que soient les valences des voisins, avec β_1 et β_2 définis sur les sommets latéraux (de valence éventuellement différente).

appelés “low frequency reconstruction masks” et “high frequency reconstruction masks” [KSS00]. Cette appellation nous semble quelque peu ambigüe, car ces masques ne servent pas directement à reconstruire le signal. Leurs coefficients se retrouvent dans les colonnes de P et de Q respectivement, alors que les masques qui sont utilisés pour la reconstruction se retrouvent dans les lignes.

Ces masques peuvent être construits comme la valeur de sortie d'une étape de synthèse (FIG. 4.8 à droite), avec en entrée des suites particulières dans $\mathbb{R}$.

i. Pour définir le masque basse fréquence, l'entrée haute fréquence est nulle, et l'entrée basse fréquence est $\delta_g = \{0, \ldots, 0, 1, 0, \ldots, 0\}$ nulle partout sauf pour un sommet d'angle $g \in \mathcal{V}^{j-1}$. En sortie on obtient un masque du type FIG. 4.9(a), avec $\beta = \beta_g$ défini par l'équation 4.4.

ii. Pour définir le masque haute fréquence, l'entrée basse fréquence est nulle, et l'entrée haute fréquence est δ_f nulle partout sauf pour un sommet d'arête $f \in \mathcal{W}^{j-1}$. En sortie on obtient un masque du type FIG. 4.9(b), avec β_1 et β_2 définis sur les deux sommets d'angle voisins, que nous noterons g et h.

Le masque haute fréquence a une grande importance : c'est lui qui définit l'ondelette. En particulier, l'ondelette a un moment nul si et seulement si la somme des coefficients de ce masque est nulle. Or cette somme vaut $1 + \frac{5}{8}(\beta_f + \beta_g) > 0$. La phase de mise à jour doit donc modifier ce masque de façon à ce que la somme soit nulle.

Premièrement, il convient de définir la forme du masque associé au filtre U. Il doit s'appliquer aux sommets d'angle $g \in \mathcal{V}^{j-1}$. Dans la mesure où l'on souhaite que ce masque soit aussi creux que possible, nous proposons un masque du même type que pour R_p, multiplié par une valeur α_g inconnue qui peut dépendre du sommet g d'application du masque (voir FIG. 4.10 à gauche). Cette valeur est le degré de liberté que nous cherchons tel que le masque haute fréquence soit de somme nulle.

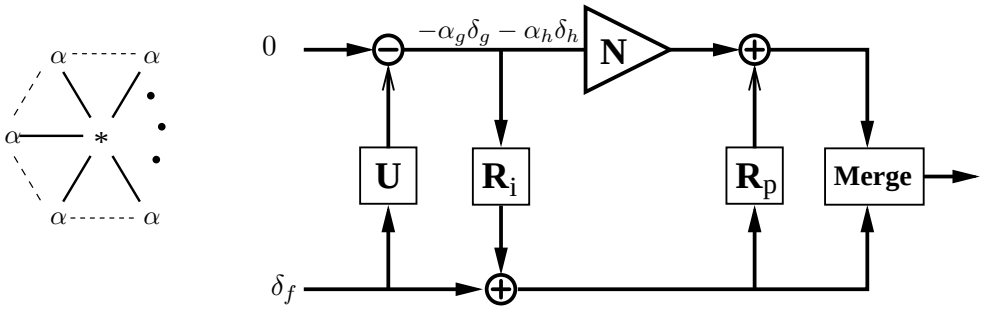


FIG. 4.10 – Calcul du filtre de mise à jour.
Masque de U (à gauche) et méthode de calcul de α (à droite).

La figure 4.10 illustre la suite de notre raisonnement. En entrée de la nouvelle synthèse (avec U), plaçons δ_f pour un sommet d'arête $f \in \mathcal{W}^{j-1}$, choisi tel que ses deux voisins immédiats g et h dans $\mathcal{V}^{j-1}$ ne sont pas des sommets de bords. L'application de U génère $-\alpha_g$ au sommet g et $-\alpha_h$ au sommet h. Autrement écrit le nouveau signal basse fréquence vaut $-\alpha_g\delta_g - \alpha_h\delta_h$.

Par linéarité, les points en sortie de la synthèse sont :

$$\begin{aligned} Q\delta_f &= \begin{bmatrix} P & Q \end{bmatrix} \begin{pmatrix} 0 \\ \delta_f \end{pmatrix} \\ &= \begin{bmatrix} \widetilde{P} & \widetilde{Q} \end{bmatrix} \begin{pmatrix} 0 \\ \delta_f \end{pmatrix} - \alpha_g \begin{bmatrix} \widetilde{P} & \widetilde{Q} \end{bmatrix} \begin{pmatrix} \delta_g \\ 0 \end{pmatrix} - \alpha_h \begin{bmatrix} \widetilde{P} & \widetilde{Q} \end{bmatrix} \begin{pmatrix} \delta_h \\ 0 \end{pmatrix} \\ &= \widetilde{Q}\delta_f - \alpha_g P\delta_g - \alpha_h P\delta_h \ , \end{aligned}$$

car $P = \widetilde{P}$. Remarquons alors que :

- la somme des coefficients du vecteur $\widetilde{Q}\delta_f$ est la somme des coefficients du masque haute fréquence (sans mise à jour) adapté en f, c'est à dire $1 + \frac{8}{5}(\beta_g + \beta_h)$,
- la somme des coefficients du vecteur $P\delta_g$ est la somme des coefficients du masque basse fréquence adapté en g, c'est à dire $1 + \frac{k_g}{2}$ (voir Fig. 4.9),
- la somme des coefficients du vecteur $P\delta_h$ est la somme des coefficients du masque basse fréquence adapté en h, c'est à dire $1 + \frac{k_h}{2}$,
- la somme des coefficients du vecteur $Q\delta_f$ est la somme des coefficients du masque haute fréquence (avec mise à jour) adapté en f, que nous voulons justement nulle.

En sommant les coefficients des vecteurs de chaque membre, l'égalité précédente implique

$$0 = \left(1 + \frac{8}{5}(\beta_g + \beta_h)\right) - \alpha_g \left(1 + \frac{k_g}{2}\right) - \alpha_h \left(1 + \frac{k_h}{2}\right).$$

Par souci de symétrie nous proposons de prendre comme coefficient pour le masque de U :

$$\alpha = \frac{5 + 16\beta}{10 + 5k}, \tag{4.5}$$

où k est la valence du sommet concerné (g ou h ici), et β est défini par l'équation (4.4). À titre d'exemple le masque haute fréquence résultant pour une portion de maillage régulière est montré 4.9(c), où l'on peut constater que la somme des coefficients est bien nulle.

Un raisonnement similaire permet de déterminer le masque de U pour les sommets de bords. Tous les résultats se trouvent sur la figure 4.6. Précisons que, les filtres U étant ainsi définis, la moyenne des sommets du maillage est conservée à la condition que celui-ci soit sans bord. Sinon la moyenne des sommets de bords est conservée (pour chaque bord du maillage pris séparément), mais pas la moyenne globale. Ceci vient du fait que

i. les courbes de bord sont entièrement définies par les points de bords, et

ii. la moyenne de tous les sommets ne dépend pas du niveau de résolution,

sont deux critères incompatibles.

Conclusion.

Nous venons de construire un schéma multirésolution paramétré sur une suite de maillages semi-réguliers. Ce schéma est basé sur la subdivision de Loop, ce qui nous assure que la surface limite est lisse (régularité des fonctions d'échelles). Les ondelettes sont d'intégrale nulle (*i.e.* la somme des coefficients du masque de haute fréquence est nulle), ce qui nous assure entre autres que la moyenne des points de contrôle est conservée d'un niveau de décomposition à

l'autre. L'utilisation de la démarche de lifting scheme nous décrit des algorithmes d'analyse et de synthèse sans allocation mémoire supplémentaire. En outre l'utilisation de masques locaux nous assure que ces algorithmes s'exécutent en temps linéaire par rapport au nombre de points de contrôle, ce qui est d'une importance capitale pour la rapidité de nos programmes.

4.3 Calcul du volume englobé par la surface

Dans la section 4.2, nous avons défini un schéma multirésolution basé sur la subdivision de Loop, qui s'applique à des surfaces paramétrées sur un maillage triangulaire semi-régulier. L'objet de cette section est de calculer, dans la base multirésolution, le volume englobé par la surface.

Au delà de l'expression mathématique du volume, notre contribution est de mettre en œuvre un algorithme efficace de calcul. Celui-ci exploite les filtres de synthèse pour pré-calculer et stocker un grand nombre de coefficients, afin d'alléger autant que possible les calculs au moment de manipuler la surface.

Dans une première section 4.3.1, nous proposons d'approximer le volume dans la base fine par le volume du maillage de contrôle fin. Il en résulte une forme tri-linéaire par rapport aux coordonnées des points de contrôle. À partir de celle-ci, nous construisons un calcul du volume dans la base multirésolution (section 4.3.2), en faisant appel à une transposition du processus de synthèse. Enfin dans la section 4.3.3, nous signalons des difficultés particulières dans l'implémentation, ayant trait aux structures de données, et nous proposons une solution.

4.3.1 Mesure du volume en base fine

Soit $S(t)$ une surface de V^n exprimée dans la base fine φ^n par la formule (4.2) :

$$S(t) = \sum_{i \in \mathcal{V}^n} \mathbf{c}_i^n \, \varphi_i^n(t) \ , \quad \text{pour } t \in \mathcal{M}^0 \ .$$

Son maillage de contrôle est $\mathbf{c}^n(\mathcal{M}^n)$. Dans le domaine d'application qui nous intéresse en premier lieu, à savoir l'animation, c'est généralement ce maillage fin qui est manipulé et représenté, et non la surface limite. Pour cette raison, nous proposons de calculer le volume de $\mathbf{c}^n(\mathcal{M}^n)$, plutôt que celui de la surface limite S.

Notons $X^n = (x_i^n)_{i \in \mathcal{V}^n}$, $Y^n = (y_i^n)_{i \in \mathcal{V}^n}$ et $Z^n = (z_i^n)_{i \in \mathcal{V}^n}$ les vecteurs coordonnées du vecteur de points de contrôle $\mathbf{c}^n$, *i.e.* $\mathbf{c}^n = (X^n, Y^n, Z^n) \in \mathbb{R}^N \times \mathbb{R}^N \times \mathbb{R}^N$. Pour calculer volume $\Theta(X^n, Y^n, Z^n)$ englobé par le maillage, nous allons raisonner avec chacune des faces qui la composent :

Soit (i, j, k) une face (orientée) de $\mathcal{M}^n$. La face correspondante du maillage géométrique $\mathbf{c}^n(\mathcal{M}^n)$ est $(\mathbf{c}_i^n, \mathbf{c}_j^n, \mathbf{c}_k^n)$. Cette face se projette dans le plan xy en un triangle

$$\begin{pmatrix} x_i^n \\ y_i^n \\ 0 \end{pmatrix} , \begin{pmatrix} x_j^n \\ y_j^n \\ 0 \end{pmatrix} , \begin{pmatrix} x_k^n \\ y_k^n \\ 0 \end{pmatrix} ,$$

dont l'aire (signée) est

$$\frac{1}{2} \begin{pmatrix} x_j^n - x_i^n \\ y_j^n - y_i^n \end{pmatrix} \otimes \begin{pmatrix} x_k^n - x_i^n \\ y_k^n - y_i^n \end{pmatrix} \ .$$

Par conséquent le volume (signé) du prisme compris entre la face et sa projection sur le plan xy est

$$\frac{z_i^n + z_j^n + z_k^n}{3} \left(\frac{1}{2} \begin{pmatrix} x_j^n - x_i^n \\ y_j^n - y_i^n \end{pmatrix} \otimes \begin{pmatrix} x_k^n - x_i^n \\ y_k^n - y_i^n \end{pmatrix} \right) .$$

Le maillage fin étant fermé et orienté, son volume s'écrit comme la somme sur les faces

$$\Theta(X^n, Y^n, Z^n) = \sum_{(i,j,k) \in \mathcal{M}^n} \frac{z_i^n + z_j^n + z_k^n}{3} \left(\frac{1}{2} \begin{pmatrix} x_j^n - x_i^n \\ y_j^n - y_i^n \end{pmatrix} \otimes \begin{pmatrix} x_k^n - x_i^n \\ y_k^n - y_i^n \end{pmatrix} \right) .$$

Remarquons que ce volume est signé, c'est-à-dire positif si la normale est orientée vers l'extérieur, négatif sinon. Cette expression est tri-linéaire par rapport aux coordonnées, et peut donc s'écrire

$$\Theta(X^n, Y^n, Z^n) = \sum_{i,j,k} \theta^n_{i\,j\,k} x_i^n \, y_j^n \, z_k^n \qquad \text{avec} \qquad 0 \leq i, j, k < m \; . \tag{4.6}$$

En égalant les deux dernières expressions, on obtient les valeurs de $\theta^n_{i\,j\,k}$:

$$\theta^n_{i\,j\,k} = \theta^n_{j\,k\,i} = \theta^n_{k\,i\,j} = \frac{1}{6} \qquad \text{et} \qquad \theta^n_{j\,i\,k} = \theta^n_{k\,j\,i} = \theta^n_{i\,k\,j} = \frac{-1}{6}$$

si (i, j, k) est une face de $\mathcal{M}^n$, et 0 sinon.

4.3.2 Expression multirésolution du volume

L'équation (4.6) nous permet de calculer le volume lorsque S est exprimé dans la base fine. Nous allons ici montrer comment exprimer ce volume lorsque S est exprimé dans la base multirésolution $(\varphi^e, \psi^e, \ldots, \psi^{n-1})$ pour un niveau de résolution $e \in \{0, \ldots, n\}$ quelconque. Dans cette base, la surface est définie par ses coefficients $(\mathbf{c}^e, \mathbf{d}^e, \ldots, \mathbf{d}^{n-1})$, et s'exprime par l'équation (4.3) :

$$S(t) = \sum_{i \in \mathcal{V}^e} \mathbf{c}_i^e \, \varphi_i^e(t) + \sum_{j=e}^{n-1} \sum_{i \in \mathcal{W}^j} \mathbf{d}_i^j \, \psi_i^j(t)$$

L'application du banc de filtres, qui conduit aux coefficients multirésolution depuis les points de contrôle fins $\mathbf{c}^n$, est une succession de combinaisons linéaires. Par conséquent nous savons que le volume a une expression tri-linéaire également dans cette base :

$$\Theta(X^e, Y^e, Z^e) = \sum_{i,j,k} \theta^e_{i\,j\,k} x_i^e \, y_j^e \, z_k^e \qquad \text{avec} \qquad 0 \leq i, j, k < m \; , \tag{4.7}$$

où $X^e = (x_i^e)_{i \in \mathcal{V}^n}$, $Y^e = (y_i^e)_{i \in \mathcal{V}^n}$ et $Z^e = (z_i^e)_{i \in \mathcal{V}^n}$ sont les vecteurs coordonnées du vecteur de coefficients multirésolution $(\mathbf{c}^e, \mathbf{d}^e, \ldots, \mathbf{d}^{n-1})$. Autrement dit, se référant à la partition (4.1) de $\mathcal{V}^n$, le coefficient (x_i^e, y_i^e, z_i^e) est le point de contrôle grossier $\mathbf{c}_i^e$ si $i \in \mathcal{V}^e$ est un sommet de $\mathcal{M}^e$, et c'est le coefficient de détail $\mathbf{d}_i^j$ si $i \in \mathcal{W}^j$ pour $j \in \{e, \ldots, n-1\}$.

Les coefficients $\theta^e_{i\,j\,k}$ ne sont *a priori* pas triviaux à calculer. L'objet de cette section est de développer une méthode récursive pour les calculer à partir des valeurs $\theta^n_{i\,j\,k}$ en base fine que nous connaissons. Considérons $\theta^e = (\theta^e_{i\,j\,k})_{i\,j\,k}$ comme une matrice à 3 dimensions. Nous allons exprimer θ^e en fonction de θ^{e+1}, afin de construire un processus de calcul $\theta^n \to \theta^{n-1} \to \cdots \to \theta^e$.

Ce processus est sémantiquement le même que celui utilisé dans la section 3.3.2 pour les surfaces produit tensoriel, et illustré par la figure 3.12 page 70. Rappelons qu'il s'agit d'appliquer

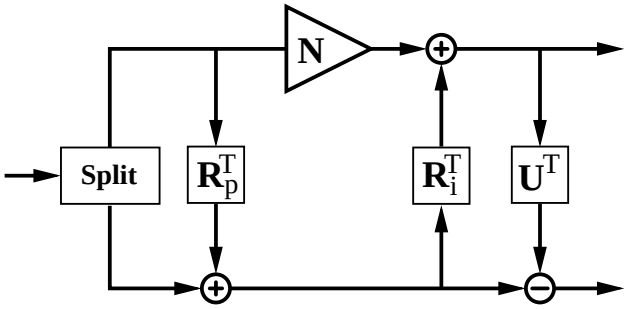


FIG. 4.11 – Lifting transposé.

le filtre de synthèse transposé $[P\,Q]^T$ sur chaque "rangée" de θ^{e+1}. Par rangée nous entendons $(\theta^{e+1}_{i\,j\,k})_i$, ou bien $(\theta^{e+1}_{i\,j\,k})_j$, ou encore $(\theta^{e+1}_{i\,j\,k})_k$.

La différence par rapport au cas produit tensoriel est dans la manière d'appliquer les filtres transposés. $[P\,Q]^T$ s'écrit sous la forme du lifting scheme transposé de la figure 4.11, ce qui se vérifie aisément si l'on se reporte à l'écriture de $[P\,Q]$ sous forme de produit de matrices. Par rapport à la synthèse (FIG. 4.8 à droite), l'ordre des phases est inversé, les filtres sont transposés, mais les opérateurs sont inchangés.

Pour appliquer ce lifting transposé, nous pouvons utiliser les masques définis pour le banc de filtres (voir FIG. 4.6), mais en "transposant" leur application. Prenons par exemple le masque de R_p FIG. 4.6(b). Dans la synthèse, son application autour d'un sommet f signifiait

$$\mathbf{c}^e_f = \mathbf{c}^e_f + \frac{5}{8}\beta_f \mathbf{d}^e_g \quad \text{pour tout } g \text{ voisin de } f.$$

Son application autour d'un sommet f dans une rangée (i,j) fixée de θ^e signifie

$$\theta^e_{i\,j\,g} = \theta^e_{i\,j\,g} + \frac{5}{8}\beta_f \theta^e_{i\,j\,f} \quad \text{pour tout } g \text{ voisin de } f.$$

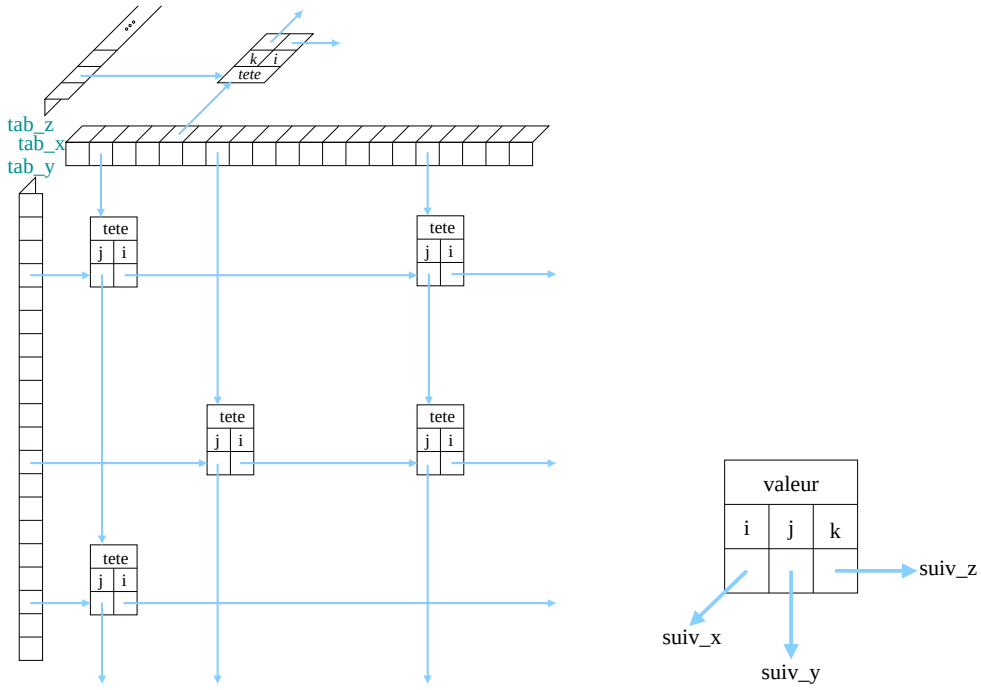
Nous venons donc de développer une méthode de calcul du volume en base multirésolution pour un niveau de résolution e quelconque (c'est à dire pour une surface définie par Eq. (4.3) page 85) constitué de trois étapes :

- Calcul et stockage de θ^n. Cette étape d'initialisation peut être faite au moment de charger la surface.
- Calcul et stockage de θ^e par la récursion $\theta^n \to \theta^{n-1} \to \cdots \to \theta^e$. Cette étape doit être effectuée pour chaque changement du niveau de résolution dans l'éditeur, à moins de les stocker pour toutes les valeurs de $e \in \{0, \ldots, n\}$.
- Calcul du volume par la formule (4.7). Seule cette troisième étape est effectuée à la volée, ce qui diminue les temps de calcul de façon importante.

4.3.3 Implémentation

Pour implémenter notre éditeur de surfaces multirésolution, nous avons choisi une structure de données basée sur un arbre quaternaire de faces inspirée de celle proposée par Certain *et al* [CPD+96]. Une telle structure permet de passer facilement d'une face à ses voisines, d'une face à ses filles, ou d'une face à sa mère. Elle est efficace aussi pour trouver les sommets d'angle encadrant un sommet d'arête. Par contre elle est assez peu pratique pour la recherche des voisins d'un sommet.

Les filtres sont codés sous forme de masques génériques (dans les parties régulières) qui sont instanciés autour d'un sommet au moment de les appliquer.



(a) Structure extérieure (b) Élément intérieur

Fig. 4.12 – Matrice tridimensionnelle creuse.

Un défi dans la mise en œuvre de notre méthode de calcul du volume est de choisir la bonne structure pour coder θ^e. En notant N le nombre de points de contrôle (*i.e.* nombre de sommets de $\mathcal{M}^n$), ces matrices sont de taille N^3. Il est donc impossible de les coder en format plein. Il nous faut un format creux qui soit suffisamment efficace sans être trop lourd, et surtout qui soit dynamique (pour pouvoir créer des éléments à l'application des filtres).

À titre d'exemple, pour un maillage d'environ 8 000 faces, la matrice θ^n a plus de 90 000 éléments non nuls, et θ^{n-1} en a environ 11×10^6. Nous conjecturons que θ^0 a $O(N \log^2 N)$ éléments non nuls, mais en pratique n (de l'ordre de $\log_4 N$) est petit, et a donc peu d'influence par rapport aux constantes multiplicatives liées à la taille des filtres.

En même temps, faute d'accéder aux éléments en temps constant (accès dit "aléatoire"), il est primordial de pouvoir parcourir rapidement les rangées de θ^e dans *n'importe quelle direction* (i, j ou k). Nous devinons ce besoin pour le calcul du volume par la formule (4.7) (chaque direction est un indice de la somme triple), ou bien pour l'application des filtres transposés (dans chaque direction séparément), et il sera encore plus pressant au vu des méthodes de déformation présentées dans la prochaine section.

Pour ces raisons nous avons choisi un triple chaînage orthogonal qui généralise la structure classique de double chaînage pour les matrices bidimensionnelles. Cette structure est illustrée figure 4.12 :

– Chaque élément regroupe sa valeur $\theta_{i\,j\,k}$, ses trois indices i, j et k, ainsi que des pointeurs

vers les éléments qui le suivent dans chaque direction (permettant de parcourir une rangée).
– Chaque rangée est accessible par sa tête, qui contient les deux indices fixes pour cette rangée, et des pointeurs pour se déplacer entre les têtes.
– Les têtes de rangées sont organisées (pour chaque paire de directions (i, j), (i, k) ou (j, k)) par un double chaînage. Sur la figure 4.12(a), on observe le double chaînage sur la "face avant", entre les têtes (i, j).

Au vu des taux de remplissage cités ci-dessus, on peut se demander s'il ne serait pas judicieux de mettre les têtes de rangées dans des tableaux bidimensionnels de taille N^2, afin d'y accéder en temps constant. Il n'est pas sûr que le gain en temps soit suffisamment important pour justifier le coût mémoire.

Au fil de l'utilisation, la manipulation de cette structure s'avère assez lourde, contrepartie inévitable de son efficacité mémoire. Pour cette raison nous envisageons de ne l'utiliser que dans la phase de construction, puis de stocker la matrice sous un format plus compact (mais statique). Nous pensons à un codage "compression de rangée", qui généraliserait les codages 2D de type compression de colonnes ou compression de lignes [Spa]. Ces formats sont plus rapides mais ont une direction privilégiée, il faudra peut être stocker l'information en triple pour conserver l'efficacité dans les trois directions.

4.4 Déformation multirésolution à volume constant

Soit S une surface multirésolution définie par (4.3). Dans la section 4.2 nous avons défini un schéma multirésolution qui permet de décomposer et reconstruire cette surface à tout niveau de résolution $e \in \{0, \ldots, n\}$. Grâce à la démarche de lifting scheme chaque étape du banc de filtres s'effectue en temps linéaire.

Dans la section 4.3 nous avons exprimé le volume englobé par cette surface dans la base multirésolution, puis nous avons construit un algorithme de calcul du volume qui tire profit de l'efficacité du lifting scheme.

L'objet de la présente section est d'intégrer le volume comme contrainte au cours d'un processus de déformation de la surface. La boucle d'édition qui schématise cette intégration est représentée figure 1.5(b) page 10 :
– La surface initiale a pour volume Θ_{ref}.
– L'utilisateur édite S à un niveau de résolution e qu'il choisit. Il en résulte une surface éditée $(\bar{X}^e, \bar{Y}^e, \bar{Z}^e)$ de volume $\bar{\Theta} = \Theta(\bar{X}^e, \bar{Y}^e, \bar{Z}^e)$.
– Nous avons à définir une surface corrigée finale (X^e, Y^e, Z^e) qui
  i. est aussi proche que possible de $(\bar{X}^e, \bar{Y}^e, \bar{Z}^e)$, et
  ii. a pour volume Θ_{ref}.

Grâce à l'expression multirésolution du volume, ce processus s'exécute entièrement sur l'expression multirésolution de la surface. Pour éditer la surface nous proposons de manipuler directement les points de contrôle $\mathbf{c}^e$, car ceci est simple et intuitif dans un outil d'édition. Néanmoins, notre méthode ne présume pas des causes de la déformation. La dernière étape de la boucle d'édition, dite étape de *correction*, est mise en œuvre grâce à une minimisation sous contrainte. La distance entre la surface déformée et la surface corrigée en constitue la fonction objectif, et la correction de volume en constitue la contrainte. Nous proposons que cette minimisation soit partielle, afin de localiser la déformation.

Linéarisation de la contrainte de volume.

Posons $(\delta_i)_{i\in\mathcal{V}^n} = (\delta X^e, \delta Y^e, \delta Z^e)$ la déformation qui doit corriger le volume, c'est-à-dire $(X^e, Y^e, Z^e) = (\bar{X}^e + \delta X^e, \bar{Y}^e + \delta Y^e, \bar{Z}^e + \delta Z^e)$, et $\delta_i \in \mathbb{R}^3$. Notre objectif est de déterminer cette déformation. La contrainte de volume s'écrit

$$\Theta(X^e, Y^e, Z^e) = \Theta(\bar{X}^e + \delta X^e, \bar{Y}^e + \delta Y^e, \bar{Z}^e + \delta Z^e) = \Theta_{ref} \ .$$

En utilisant la tri-linéarité de Θ, nous pouvons écrire

$$\begin{aligned}\Theta_{ref} &= \Theta(\bar{X}^e + \delta X^e, \bar{Y}^e + \delta Y^e, \bar{Z}^e + \delta Z^e)\\ &= \Theta(\bar{X}^e, \bar{Y}^e, \bar{Z}^e) + \Theta(\bar{X}^e, \bar{Y}^e, \delta Z^e) + \Theta(\bar{X}^e, \delta Y^e, \bar{Z}^e) + \Theta(\delta X^e, \bar{Y}^e, \bar{Z}^e)\\ &\quad + \Theta(\bar{X}^e, \delta Y^e, \delta Z^e) + \Theta(\delta X^e, \bar{Y}^e, \delta Z^e) + \Theta(\delta X^e, \delta Y^e, \bar{Z}^e) + \Theta(\delta X^e, \delta Y^e, \delta Z^e).\end{aligned}$$

Cette contrainte est cubique par rapport aux variables δX^e, δY^e et δZ^e. Pour l'intégrer plus facilement dans la minimisation, nous proposons de la linéariser en négligeant les quatre derniers termes qui sont quadratiques ou cubiques. Cette approximation présuppose que la déformation $(\delta X^e, \delta Y^e, \delta Z^e)$ est petite devant $(\bar{X}^e, \bar{Y}^e, \bar{Z}^e)$. En supposant que cette hypothèse est vérifiée, la contrainte de volume se reformule

$$\Theta_{ref} - \bar{\Theta} = \Theta(\bar{X}^e, \bar{Y}^e, \delta Z^e) + \Theta(\bar{X}^e, \delta Y^e, \bar{Z}^e) + \Theta(\delta X^e, \bar{Y}^e, \bar{Z}^e) \ . \tag{4.8}$$

Localisation de la correction de volume.

Afin de mieux contrôler la correction, il est intéressant de pouvoir la circoncire à une région bien définie de la surface. Cette région peut par exemple être définie par un voisinage du point de contrôle édité. Cela revient à définir un ensemble $\mathcal{L} \subset \mathcal{V}^n$ de sommets dits "libres". Les sommets $i \in \mathcal{L}$ sont ceux qui appartiennent à la région d'influence. Pour $i \notin \mathcal{L}$, on fixe $\delta_i = 0$. Pour prendre en compte cette modalité supplémentaire, posons

$$\Theta^e(i) = \begin{pmatrix}\Theta^e_{\bar{Y}\bar{Z}}(i)\\ \Theta^e_{\bar{X}\bar{Z}}(i)\\ \Theta^e_{\bar{X}\bar{Y}}(i)\end{pmatrix} \in \mathbb{R}^3 \quad \text{pour} \quad 0 \le i < N,$$

où les sommes partielles sont données par

$$\Theta^e_{\bar{Y}\bar{Z}}(i) = \sum_{j,k=0}^{N-1} \theta^e_{i\,j\,k}\, \bar{y}^e_j\, \bar{z}^e_k \quad \text{pour} \quad 0 \le i < N,$$

$$\Theta^e_{\bar{X}\bar{Z}}(j) = \sum_{i,k=0}^{N-1} \theta^e_{i\,j\,k}\, \bar{x}^e_i\, \bar{z}^e_k \quad \text{pour} \quad 0 \le j < N,$$

$$\Theta^e_{\bar{X}\bar{Y}}(k) = \sum_{i,j=0}^{N-1} \theta^e_{i\,j\,k}\, \bar{x}^e_i\, \bar{y}^e_j \quad \text{pour} \quad 0 \le k < N.$$

En notant "$\cdot$" le produit scalaire dans $\mathbb{R}^3$, la contrainte (4.8) se réécrit alors

$$\Theta_{ref} - \bar{\Theta} = \sum_{i\in\mathcal{V}^n} \Theta^e(i)\cdot\delta_i = \sum_{i\in\mathcal{L}} \Theta^e(i)\cdot\delta_i \ .$$

Limiter le nombre de coefficients libres permet en outre de profiter de la décomposition multirésolution. Si $\mathcal{L} \subset \mathcal{V}^e$, seuls certains coefficients d'échelles (*i.e.* des points de contrôle) sont libres, et tous les coefficients de détails ($i \in \mathcal{W}^j$, $j \in \{e, \ldots, n-1\}$) sont fixes. Cela permet de ne modifier que la forme globale de la surface, en conservant les détails qui codent les caractéristiques locales. En outre la correction sera beaucoup moins coûteuse à calculer.

Minimisation sous contrainte pour définir la correction.

Une fois définie l'étendue $\mathcal{L}$ de la correction, il nous reste à formuler la résolution du problème. Nous proposons une minimisation de la norme quadratique du déplacement, afin de minimiser l'écart entre la surface finale et la surface éditée :

$$\min_{\delta_i} \sum_{i\in\mathcal{L}} \|\delta_i\|^2 \quad \text{sous contrainte} \quad \sum_{i\in\mathcal{L}} \Theta^e(i)\cdot\delta_i = \Theta_{ref} - \bar{\Theta}.$$

En utilisant un multiplicateur de Lagrange λ, cela revient à trouver un point stationnaire $\vec{\bigtriangledown} g = 0$ du lagrangien

$$g\big((\delta_i)_{i\in\mathcal{L}},\lambda\big) = \sum_{i\in\mathcal{L}} \|\delta_i\|^2 + \lambda\Big(\sum_{i\in\mathcal{L}} \Theta^e(i)\cdot\delta_i - (\Theta_{ref}-\bar{\Theta})\Big)\ .$$

Ce point stationnaire est solution du système linéaire

$$\begin{cases} 2\delta_i + \lambda\Theta^e(i) &= 0 & \forall i\in\mathcal{L}\ , \\ \sum \Theta^e(i)\cdot\delta_i &= \Theta_{ref}-\bar{\Theta} & \text{somme sur } i\in\mathcal{L}. \end{cases}$$

Et la solution se formule explicitement par :

$$\lambda = \frac{-2(\Theta_{ref}-\bar{\Theta})}{\sum_j \|\Theta^e(j)\|^2}$$

et

$$\delta_i = \frac{\Theta_{ref}-\bar{\Theta}}{\sum_j \|\Theta^e(j)\|^2}\ \Theta^e(i) \quad \text{pour } i\in\mathcal{L}. \tag{4.9}$$

Analyse algorithmique.

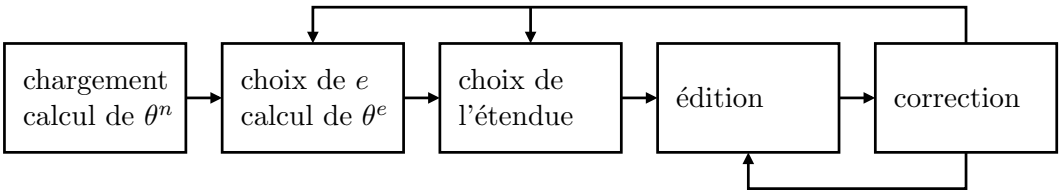


FIG. 4.13 – Processus de déformation.

Nous avons maintenant une vue d'ensemble du processus de déformation, illustré figure 4.13, tel qu'il intervient dans l'outil d'édition. Soit $N = card(\mathcal{V}^n)$ le nombre de sommets du maillage fin $\mathcal{M}^n$, et $L = card(\mathcal{L})$ le nombre de coefficients libres dans la déformation. Le processus est composé de de cinq étapes :

i. La surface est chargée dans l'éditeur, et la matrice initiale θ^n est pré-calculée d'après les résultats de la section 4.3. Son coût est $O(N)$ car le schéma multirésolution que nous avons choisi est local. Ceci ne comprend pas la construction de la structure de données pour stocker la surface. À ce moment peuvent aussi être construits les masques intervenant dans le lifting scheme FIG. 4.8.

ii. L'utilisateur détermine le niveau de résolution e auquel il souhaite éditer la surface. Cette dernière est décomposée à l'aide du banc de filtres en temps linéaire pour chaque niveau de résolution. La matrice θ^e est calculée à partir de θ^n par le processus récursif défini dans la section 4.3, à l'aide du lifting transposé Fig. 4.11.
Le coût de cette dernière opération est difficile à évaluer car il faudrait connaître précisément le taux de remplissage de θ^j, $j \in \{e, \dots, n\}$. Faute de mieux, nous pouvons affirmer que le coût de chaque étape $\theta^j \rightarrow \theta^{j-1}$ est $O((card\,\mathcal{V}^j)^3)$. Par conséquent le coût total du processus récursif est $O(N^3)$, ce qui n'est probablement pas la borne asymptotique optimale. Nous avons constaté en pratique que cette étape est en effet lourde (plusieurs minutes pour $N = 8\,000$), ce qui nous incite à envisager un pré-calcul de toutes les matrices θ^e au moment du chargement, voire un stockage sur disque.

iii. Une région d'influence est déterminée. Le coût de cette opération est en principe assez faible, et dépend de la façon dont elle est traitée. Il est raisonnable de la considérer en $O(L)$, ce qui est par exemple le cas si l'on utilise un critère de distance maximale sur le maillage.

iv. L'utilisateur édite la surface par déplacement d'un sommet du maillage de contrôle $\mathbf{c}^e(\mathcal{M}^e)$ par une opération de *drag-and-drop*.

v. Le volume est corrigé par une déformation de la région $\mathcal{L}$, selon l'équation (4.9). Cette étape est la seule à être effectuée pour chaque déformation.
Les deux opérations coûteuses sont le calcul des $\Theta^e(i)$ en $O(LN)$, et le calcul du volume en $O(N^3)$. Cette dernière estimation est largement surévaluée, car elle a en fait un coût équivalent au nombre d'éléments non nuls de θ^e, que nous ne connaissons pas, mais que nous conjecturons être $O(N \log^2 N)$.
Après cette correction, soit l'utilisateur continue à éditer le même point de contrôle, dans quel cas on retourne à l'étape (iv), soit il choisit d'éditer un autre point de contrôle à cette même échelle, dans quel cas on retourne à l'étape (iii), soit il choisit de modifier le niveau de résolution, dans quel cas on retourne à l'étape (ii).

Contraintes additionnelles.

Il est possible d'incorporer des contraintes linéaires supplémentaires à notre processus de déformation, en les intégrant comme contraintes pour la minimisation. Il peut être par exemple intéressant d'ajouter des contraintes de position [Sta98], de tangence ou de normale. La difficulté est que le système linéaire ne sera plus nécessairement solvable explicitement, ce qui nous obligerait à utiliser une méthode de résolution générale de système creux. La résolution sera d'autant plus coûteuse.

4.5 Résultats

Pour illustrer notre méthode, observons la figure 4.14, qui représente une balle qui chute et se déforme en arrivant au sol. Les maillages fins $\mathbf{c}^2(\mathcal{M}^2)$ ont 320 faces et 162 sommets. Le maillage de contrôle $\mathbf{c}^0(\mathcal{M}^0)$ est un icosaèdre (20 faces, 12 sommets). Chaque ligne de la figure représente une simulation, avec différents paramètres :

(a) Une balle lisse, déformée *sans* conservation du volume.
(b) La même balle lisse, déformée *avec* conservation du volume.
(c) Une balle à laquelle des détails ont été ajoutés à une échelle fine, déformée *avec* conservation du volume.

Ces simulations ont été créées interactivement, toutes trois de la même manière, par déplacement de quatre points de contrôle (sommets de $\mathbf{c}^0(\mathcal{M}^0)$). La comparaison des deux premières lignes nous montre l'intérêt de la conservation du volume : cela donne l'impression d'une balle qui se déforme de façon réaliste, sans diminuer de taille quand elle s'écrase au sol. La ligne du bas illustre l'intérêt du modèle multirésolution : les détails fins sont conservés au cours du processus de déformation, sans pour autant complexifier la manipulation de la surface, *i.e* grâce aux mêmes 4 sommets de $\mathbf{c}^0(\mathcal{M}^0)$.

La figure 4.15 présente trois déformations à volume constant du même maillage initial, représenté en haut (a), avec et sans maillage de contrôle. Le maillage fin $\mathbf{c}^2(\mathcal{M}^2)$ comporte 1732 sommets et 3520 faces, et le maillage de contrôle $\mathbf{c}^0(\mathcal{M}^0)$ comporte 112 sommets et 220 faces. Les trois déformations qui suivent ont été créées en déplaçant les points de contrôle de $\mathbf{c}^0(\mathcal{M}^0)$:

(b) La patte avant gauche est pliée, en déplaçant seulement 6 points de contrôle.

(c) La tête est baissée et le cou est tendu vers l'avant, en conservant le volume localement (de la tête jusqu'à l'encolure).

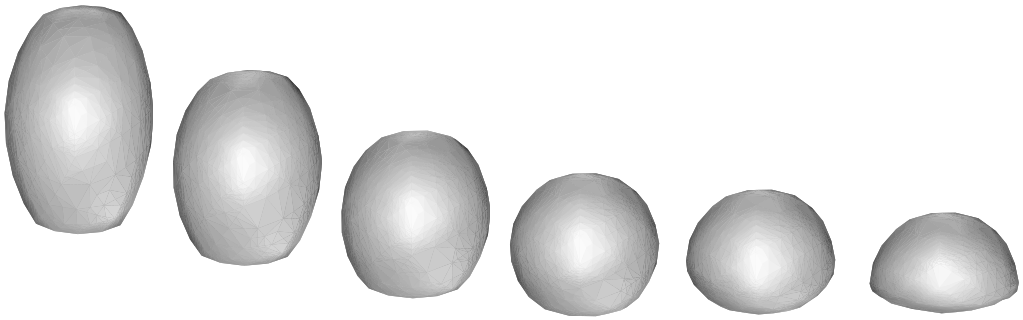
(d) Pour cabrer le cheval, un plus grand nombre de points de contrôle sont modifiés, *i.e.* une accumulation de plusieurs défomations. Pour chaque déformation, le volume est conservé localement, afin de préserver les proportions entre les différentes parties du corps.

Grâce à la combinaison de la minimisation (de la norme quadratique de la déformation) et de la contrainte locale de volume, le maillage est aisément déformé avec réalisme. La norme quadratique favorise de petits déplacements d'un grand nombre de points, plutôt qu'une modification importante portant sur peu de points, ce qui limite les distorsions. En même temps, la contrainte de volume permet de conserver les proportions de chaque partie du maillage. En outre, la représentation multirésolution permet de conserver les détails du maillage sans avoir à les manipuler, et donc de conserver l'apparence locale de la surface.

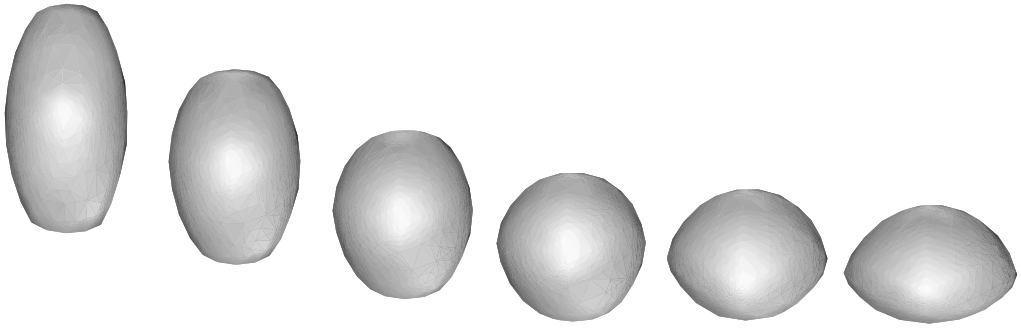
La figure 4.16 présente une déformation des oreilles du *Stanford's bunny*. Le volume est conservé localement, ce qui permet de conserver la taille des oreilles, tout en déplaçant librement les points de contrôle pour créer la déformation.

Tous les exemples qui ont été présentés dans cette section ont été créés par manipulation interactive des points de contrôle dans un éditeur graphique multirésolution codé par nos soins. Lors de l'édition d'un point de contrôle, le calcul de la déformation à volume constant prend moins de 100 ms sur un Pentium 4 cadencé à $2,6$ GHz. Le temps de calcul ne pose donc aucun problème pour des maillages de quelques milliers de points de contrôle.

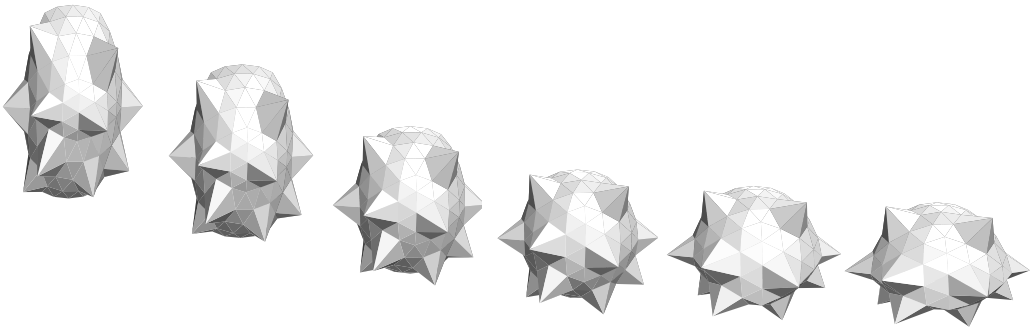
A contrario, la place mémoire est un facteur limitant puisque, par exemple pour le maillage de la figure 4.15, la place occupée est de l'ordre de 200 Mo. Ceci est malheureusement une limitation inhérente au calcul de volume. Il n'est pas possible de stocker moins que les éléments non nuls, répertoriés par notre structure de données creuse à trois dimensions.



(a) Sans conservation du volume.

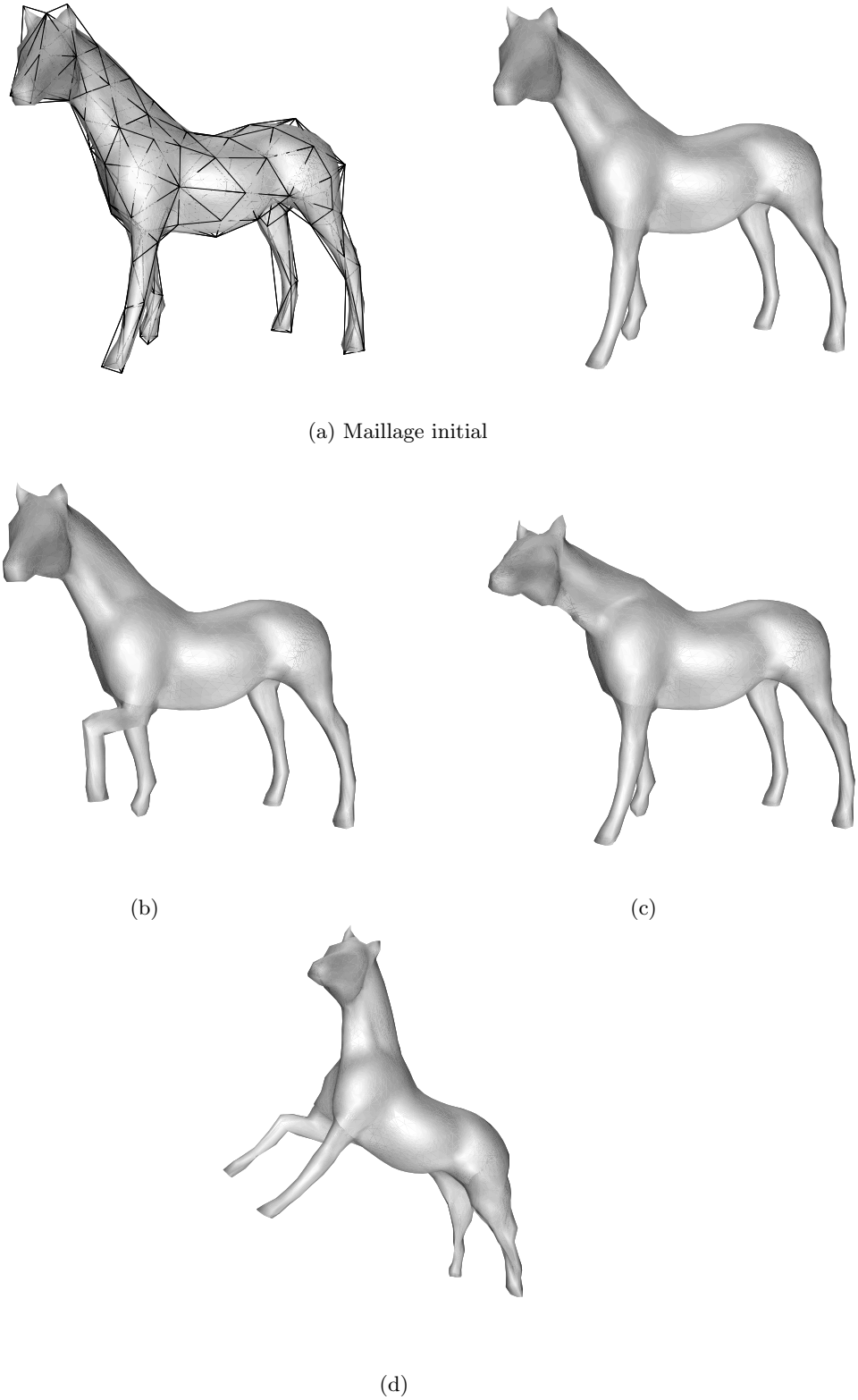


(b) Avec conservation du volume.



(c) Avec conservation du volume.

Fig. 4.14 – Balle qui chute.
La ligne (a) représente une balle lisse qui se déforme *sans* conservation de volume.
La ligne (b) représente la même balle lisse qui se déforme *avec* conservation de volume.
La ligne (c) représente une balle avec des détails fins, qui se déforme *avec* conservation de volume, tout en gardant ses détails. Ce maillage est visualisé en *flat shading*.
Les maillages fins ont 162 sommets et 320 faces. Le maillage de contrôle est un icosaèdre.



(a) Maillage initial

(b)

(c)

(d)

FIG. 4.15 – Animation d'un cheval.
Sur la ligne du haut, le maillage initial (3520 faces) est représenté avec et sans le maillage de contrôle (220 faces). Suivent trois déformations à volume constant : la patte avant gauche (b), la tête jusqu'à l'encolure (c), puis une déformation plus globale (d). Maillage mis à disposition par *Caltech Multi-Res Modeling Group* [MRMG].

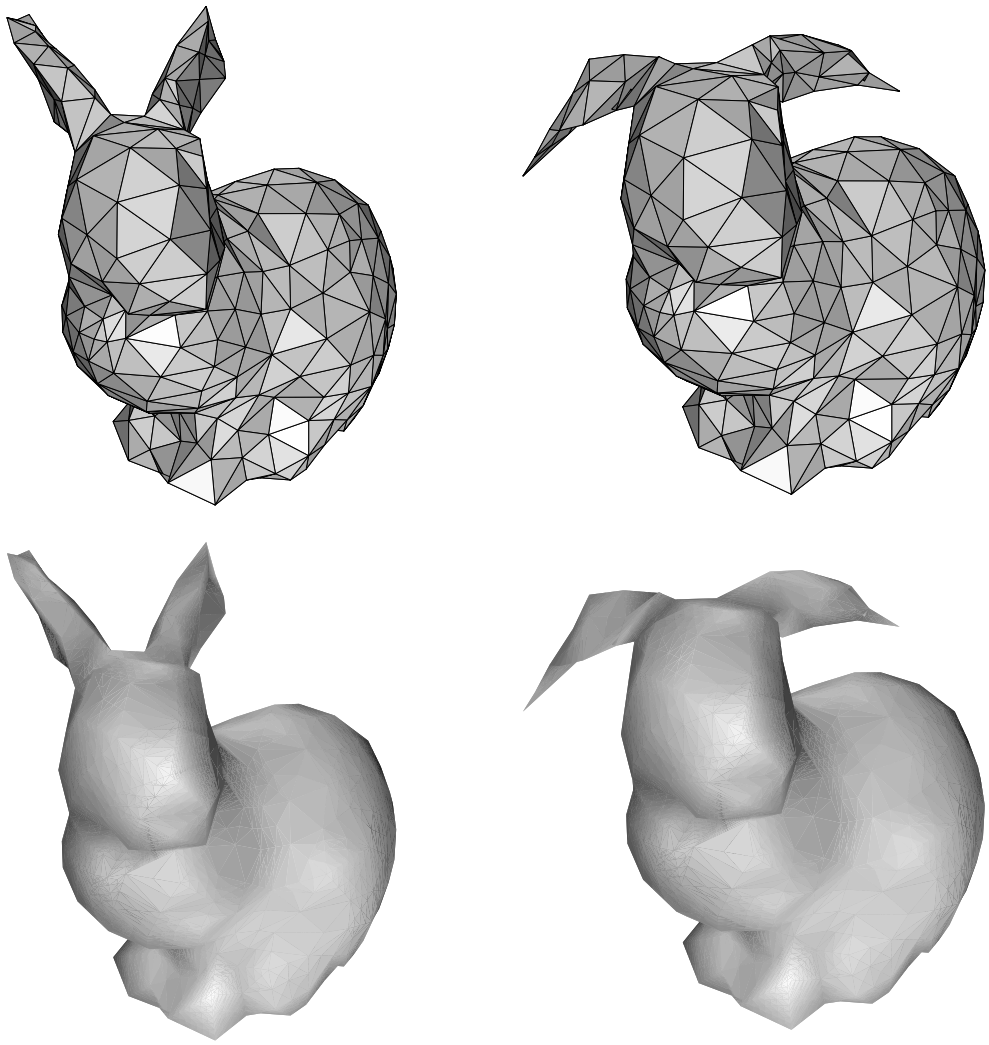


Fig. 4.16 – Animation du lapin de Stanford.
Le maillage fin comporte 1000 faces et 502 sommets. Les deux mêmes positions des oreilles sont représentées en haut avec les arêtes du maillage fin, en bas avec un *Gouraud shading*. Maillage mis à disposition par *Caltech Multi-Res Modeling Group* dans sa version semi-régulière [MRMG], d'après un modèle initial de l'université de Stanford.

Chapitre 5

Longueur et courbes tridimensionnelles

Dans ce chapitre nous présentons une méthode de déformation de courbes linéaires 3D à longueur constante. Le principal apport de ces travaux est la méthode d'intégration de cette contrainte dans un environnement d'édition multirésolution, permettant un contrôle intuitif de l'étendue et de l'aspect de la déformation. L'intégration se fait en deux étapes centrées autour de la représentation multirésolution. Par la suite, ce processus est mis au cœur d'un outil de modélisation surfacique de tissus mous, pour générer des plis sur un maillage : une courbe extraite du maillage est déformée à longueur constante, fournissant un profil de plis qui est propagé sur un voisinage. Nous montrons par là que la contrainte unidimensionnelle permet de simuler avec efficacité et réalisme un comportement bidimensionnel.

5.1 Problématique

La conservation de longueur lors de déformations est un principe de réalisme en animation remarqué de longue date [Las87]. Par exemple, pour animer des tissus mous incompressibles comme les vêtements ou la peau, la contrainte de longueur peut imiter avantageusement des lois physiques [BW98, DSB99]. De tels tissus se caractérisent entre autres par leur propension à former des plis lorsqu'ils sont comprimés. Les plis se forment pour compenser la perte de longueur induite par la déformation.

Il existe un certain nombre de modèles dédiés à la création de plis dans le domaine de l'animation. Une première catégorie peut regrouper les méthodes basées sur le bump mapping, introduit par Blinn [Bli78] en 1978 puis largement réutilisé. Le bump mapping consiste à modifier les normales à la surface de façon à donner l'illusion de plis au moment du rendu avec éclairage. Le principal défaut de cette méthode est qu'elle ne modifie pas la géométrie de la surface. Le profil de la surface ne change pas, donc le rendu n'est vraiment probant que pour de petites déformations, et si le point de vue est quasi orthogonal à la surface.

Par ailleurs, certains travaux se consacrent à la simulation des plis *statiques* de la peau, en particulier les travaux de Thalmann et ses collaborateurs [WKMT96, BKMTK00]. Par plis statiques nous entendons des rides attachées au maillage, qui ne résultent pas de la déformation de celui-ci. Un certain nombre d'autres travaux traitent la question des plis *dynamiques*, c'est-à-dire des plis qui se font et se défont au gré des déformation du maillage. Beaucoup de ces méthodes [VY92, WKMT96, WKMMT99] concernent l'animation faciale, et nécessitent un dessin manuel préalable des lignes de rides. Ces méthodes sont bien appropriées pour simuler des expressions de visage, mais elles restent très spécifiques. Nous allons ici plutôt développer une méthode générale basée sur une contrainte géométrique, et ne présume pas des particularités de l'objet déformé.

Dans [HBVMT99], Hadap *et al* développent une méthode de simulation de plis dynamiques sur des maillages. Une "displacement map" est construite dans l'objectif de conserver l'aire du maillage : la déformation sous-jacente d'un maillage grossier sert à moduler des motifs de plis prédéfinis. Cette méthode est convaincante pour le rendu de vêtements, mais elle a des inconvénients pour une utilisation plus générale : les motifs doivent être dessinés par l'utilisateur, et surtout la géométrie est modifiée au moment du rendu par une couche de "displacement mapping" qui s'ajoute à l'animation du maillage. Par conséquent elle ne peut être utilisée dans pour générer des surfaces qui seront ensuite animées.

Jing *et al* [JJT05] ont récemment présenté un outil de création de plis pour la conception de chaussures. Cet outil est basé sur la propagation d'un profil de courbes défini au bord de patchs Nurbs. Au vu de nos propres objectifs, cette méthode utilise une étape rédhibitoire de triangulation qui rend impossible toute utilisation dynamique. Suite à cette étape, la représentation Nurbs de la surface est perdue, donc il est impossible d'appliquer plusieurs déformations successives.

Larboulette et Cani [LC04] ont construit une méthode de création de plis sur des maillages basée sur la propagation d'une courbe plane de profil. Il est possible d'y superposer des plis à différentes échelles. Ces travaux sont donc proches de ce que nous allons présenter ici. Nous noterons deux principaux désavantages. Tout d'abord, le profil de plis est créé indépendamment de la géométrie du maillage, ce qui fait que les plis sont très réguliers. Deuxièmement leur méthode de propagation fait intervenir la projection sur un plan médian, ce qui pose problème dans les régions de forte courbure.

Le problème que nous nous proposons d'aborder ici est celui de la déformation de courbes tridimensionnelles et bidimensionnelles linéaires par morceaux avec conservation de leur longueur (section 5.2). La difficulté à traiter la contrainte de longueur provient du fait que cette contrainte n'est pas seulement multi-linéaire comme l'aire ou le volume, mais plus complexe. Nous proposons d'intégrer la contrainte de longueur dans le processus de déformation par deux étapes successives, chacune opérant à un niveau de résolution différent. La première étape approxime la contrainte à une échelle intermédiaire. La deuxième étape consiste en une optimisation à l'échelle la plus fine pour satisfaire la contrainte avec exactitude tout en lissant la courbe. La représentation multirésolution a ici deux intérêts : d'une part elle permet d'éditer la courbe facilement à une large échelle, et d'autre part le choix de l'échelle intermédiaire pour la première étape permet de contrôler la taille des plis. Nous montrons ensuite (section 5.3) comment intégrer cette contrainte dans un outil de manipulation de maillages de façon à y créer des plis en contrôlant l'étendue de la déformation. Nous commençons pour cela par extraire une courbe sur la surface, puis nous la déformons avant de la réintégrer comme profil de plis en l'atténuant transversalement sur une région pré-définie du maillage.

5.2 Déformation de courbes 3D à longueur constante

5.2.1 Courbes multirésolution linéaires par morceaux

Nous travaillons dans le cadre multirésolution présenté en section 1.3. Les espaces emboîtés V^j contiennent des courbes ouvertes linéaires par morceaux à 2^j segments, ou courbes polygonales, paramétrées sur l'intervalle $[0,1] \in \mathbb{R}$. Elles sont définies par 2^j+1 sommets dans $\mathbb{R}^3$. Une courbe $\mathbf{c} \in V^n$ s'exprime comme :

$$\mathbf{c}(t) = (x(t), y(t), z(t)) = \sum_{i=0}^{2^n} \mathbf{c}_i^n \ \varphi_i^n(t) = (\mathbf{c}^n)^T \varphi^n(t), \ \ t \in [0,1],$$

où $\mathbf{c}^n = (\mathbf{c}_0^n, \mathbf{c}_1^n, ..., \mathbf{c}_{2^n}^n)^T \in (\mathbb{R}^3)^{2^n+1}$ est le vecteur colonne des points de contrôle de $\mathbb{R}^3$, c'est-à-dire des sommets de la courbe polygonale. Les fonctions de base $\varphi_i^n(t)$ sont les fonctions chapeau, ou B-spline de degré 1. $\varphi^n = (\varphi_i^n)_i$ est le vecteur colonne regroupant ces fonctions.

Soit e un niveau de résolution quelconque entre 0 et n. L'espace V^n se décompose en (cf. Eq. (1.2) page 5)

$$V^n = V_e \bigoplus_{j=e}^{n-1} W_j.$$

Cette décomposition est munie d'une base
- de fonctions d'échelles $\varphi_i^e(t)$, $i \in \{0, \dots, 2^e\}$ (base de V^e), et
- d'ondelettes ψ_i^j, $i \in \{0, \dots, 2^j - 1\}$, pour chaque échelle $j \in \{e, \dots, n-1\}$ (bases de W^j).

Une courbe $\mathbf{c}(t) \in V^n$ s'exprime dans cette base comme combinaison linéaire des fonctions d'échelle φ^e et des ondelettes ψ^j :

$$\mathbf{c}(t) = (\mathbf{c}^e)^T \varphi^e(t) + (\mathbf{d}^e)^T \psi^e(t) + \cdots + (\mathbf{d}^{n-1})^T \psi^{n-1}(t), \quad e = 0, \dots, n. \tag{5.1}$$

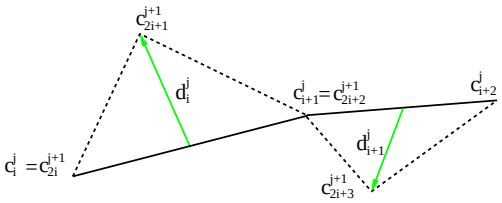


FIG. 5.1 – Schéma linéaire interpolant.
Le polygone de contrôle au niveau j est en trait plein, au niveau $j+1$ en pointillés.

Dans ce schéma multirésolution, appelé schéma linéaire interpolant, les filtres de décomposition et de recomposition s'écrivent (voir FIG. 5.1) :

$$\text{décomposition} \quad \begin{cases} \mathbf{c}_i^j = \mathbf{c}_{2i}^{j+1} \\ \mathbf{d}_i^j = \mathbf{c}_{2i+1}^{j+1} - \frac{1}{2}(\mathbf{c}_{2i}^{j+1} + \mathbf{c}_{2i+2}^{j+1}) \end{cases} \tag{5.2}$$

$$\text{recomposition} \quad \begin{cases} \mathbf{c}_{2i}^{j+1} = \mathbf{c}_i^j \\ \mathbf{c}_{2i+1}^{j+1} = \frac{1}{2}(\mathbf{c}_i^j + \mathbf{c}_{i+1}^j) + \mathbf{d}_i^j \ . \end{cases} \tag{5.3}$$

Il nous faut justifier ici le choix de ce schéma qui peut sembler simpliste :
- Il est interpolant, c'est-à-dire que les points de contrôle $\mathbf{c}_i^j$ appartiennent à la courbe. Cela nous permet, en manipulant les points de contrôle, de manipuler directement la courbe, ce qui facilite l'édition.
- Comme nous le verrons dans la suite de cette section, ceci nous permet d'exprimer les contraintes de longueur sous forme quadratique. L'algorithme de déformation (section 5.2.2) pourra alors atteindre l'objectif temps réel.
- L'application phare est la création de plis sur des maillages quelconques (section 5.3), et les courbes extraites d'un maillage, que nous utilisons en entrée, sont polygonales.

En contrepartie, ce schéma est peu lisse, ce qui nous oblige à prendre en compte un critère de lissage dans nos algorithmes. L'impossibilité de gérer des courbes analytiquement régulières n'est pas très restrictive en soi pour des applications en informatique graphique, dans la mesure où un polygone suffisamment fin apparaît visuellement à l'écran comme une courbe lisse.

Notations.

Fixons les notations utilisées dans ce chapitre :
- $\mathbf{c}_i = (x_i, y_i, z_i) \in \mathbb{R}^3$ sont les points de contrôle ;
- la courbe $\mathbf{c}(t)$ est définie au niveau j par ses coefficients d'échelle $\mathbf{c}^j$ et ses coefficients d'ondelette $\mathbf{d}^j, \ldots, \mathbf{d}^{n-1}$, tous concaténés dans un vecteur colonne noté $C^j \in (R^3)^{2^n+1}$, $j \in \{0, \ldots, n\}$;
- C^j est donc un vecteur de $2^n + 1$ coefficients dans $\mathbb{R}^3$ qui sera décomposé selon trois composantes dans $\mathbb{R}^{2^n+1}$, notées X^j, Y^j et Z^j.

Longueur d'une courbe polygonale.

Dans le cas général d'une courbe $\mathbf{c}(t) = (x(t), y(t), z(t))$ continue et dérivable par morceaux, la longueur est

$$L = \int \sqrt{x'(t)^2 + y'(t)^2 + z'(t)^2} dt \ .$$

Dans notre cas (une courbe linéaire par morceaux) elle se simplifie en

$$L = \sum_{i=0}^{2^n-1} ||\mathbf{c}_{i+1}^n - \mathbf{c}_i^n||_2 \ . \tag{5.4}$$

Cette équation exprime la longueur totale de la courbe, et pourrait a priori servir telle quelle de contrainte pour le processus de déformation. Nous allons plutôt choisir de conserver la longueur de chaque segment séparément, et ce pour deux raisons :

- Cela va nous permettre d'exprimer les contraintes sur les carrés des longueurs, et ainsi faire disparaître les racines carrées.
- On conserve l'équilibre entre les longueurs des segments, évitant ainsi que les points de contrôle se concentrent sur une portion de la courbe en laissant le reste clairsemé.

Les contraintes de longueur sur la courbe fine C^n (qui est égal à $\mathbf{c}^n$) s'expriment alors comme

$$f_i(C^n) = \Delta x_i^2 + \Delta y_i^2 + \Delta z_i^2 - l_i^2 = 0 \quad \text{pour} \quad i = 0, \ldots, 2^n - 1 \ , \tag{5.5}$$

où l_i est la longueur de référence du segment $[\mathbf{c}_i^n; \mathbf{c}_{i+1}^n]$ (*i.e.* avant déformation), $\Delta x_i = x_{i+1}^n - x_i^n$, $\Delta y_i = y_{i+1}^n - y_i^n$ et $\Delta z_i^n = z_{i+1}^n - z_i^n$.

5.2.2 Principe de la déformation à longueur constante

Les contraintes de longueur étant exprimées par (5.5), expliquons maintenant comment les intégrer dans un processus de déformation de courbes 3D. Cette intégration est résumée par la boucle d'édition Fig. 5.2 que nous expliquons maintenant.

La motivation de cette solution est de mettre la représentation multirésolution au cœur du processus pour en exploiter au mieux les capacités géométriques :

i. Comme nous l'avons souligné dans l'introduction, elle permet de manipuler la courbe aisément, de la déformer globalement avec seulement quelques points de contrôle, tout en conservant les détails.

ii. L'utilisation de plusieurs niveaux de décomposition va nous permettre de contrôler précisément l'échelle de déformation à chaque étape du processus. En particulier pour la création de plis, nous souhaitons contrôler la taille des plis grâce à la décomposition à une échelle spécifique.

Quand l'utilisateur édite la courbe, la correction de la longueur se fait en deux étapes, développées dans les sections 5.2.3 et 5.2.4. La première a pour rôle de créer la structure des plis à une échelle intermédiaire tout en approximant la longueur. La deuxième a pour rôle de satisfaire exactement les contraintes (5.5) tout en lissant la courbe.

Trois *niveaux de décomposition* (correspondant aux trois colonnes FIG. 5.2) entrent en jeu :
- n est le niveau le plus fin,
- e est le niveau d'édition,
- w est un niveau intermédiaire correspondant à l'échelle des plis éventuellement induits par la correction de longueur.

Quatre *états* de la courbe apparaissent dans la boucle d'édition :
- C_R est la courbe initiale, aussi courbe de *référence* pour la longueur,
- C_D est la courbe *déformée*, c'est-à-dire après avoir été éditée par l'utilisateur, mais avant que la longueur soit corrigée,
- C_A est la courbe *attractive*, issue de l'étape de conservation explicite de la longueur,
- C_F est la courbe *finale* qui remplacera C_R pour la prochaine boucle d'édition.

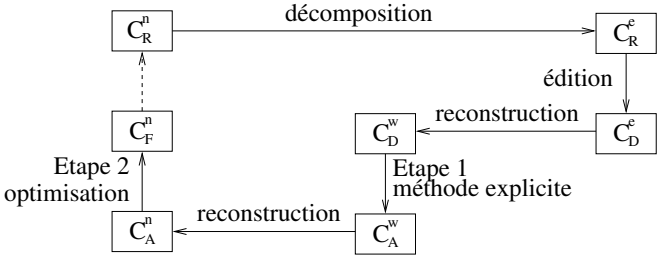


FIG. 5.2 – Boucle d'édition.
Chaque boite représente un état de la courbe à un certain niveau de décomposition, chaque transition horizontale un changement de niveau de décomposition (n, e et w), et chaque transition verticale un changement d'état de la courbe (R, D, A et F).

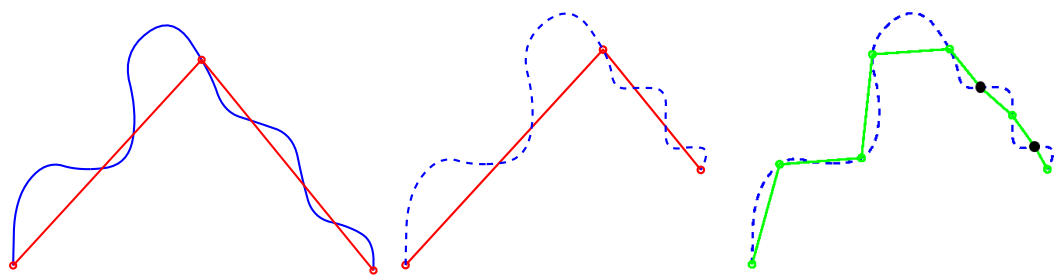
Décrivons maintenant avec plus de précisions le rôle de chaque transition de la boucle FIG. 5.2 en nous appuyant sur l'illustration FIG. 5.3 :

Décomposition : La courbe est décomposée en C_R^e au niveau e choisi par l'utilisateur avec les formules (5.2). Le choix de e détermine l'étendue de la déformation : si $e = 0$ la courbe entière sera modifiée, et plus e est grand plus la déformation est locale. Sur la figure 5.3(a) il y a 3 points de contrôle, $e = 1$.

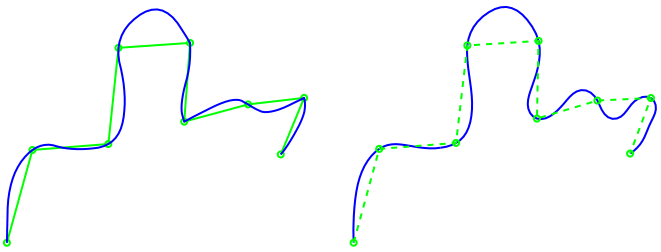
Édition : Le polygone de contrôle $\mathbf{c}_R^e$ de C_R^e est modifié par l'utilisateur. La courbe déformée résultante $\mathbf{c}_D(t)$ n'a pas la même longueur que la courbe de référence $\mathbf{c}_R(t)$. Cette déformation, qui n'est que la première étape du processus complet, est une déformation multirésolution classique. Sur la figure 5.3(b) la moitié droite de la courbe est compressée.

Reconstruction : La courbe déformée est partiellement reconstruite au niveau w ($e \leq w < n$), choisi comme définissant l'échelle des plis. Sur la figure 5.3(c) on observe le polygone de contrôle au niveau $w + 1$: la courbe fine n'a pas changé, mais le niveau de décomposition a augmenté.

Conservation explicite de la longueur : Une courbe attractive C_A est construite. Elle est obtenue par modification de C_D^w au niveau w en utilisant les détails $\mathbf{d}^w$ de façon à ce que les arêtes du polygone de contrôle $\mathbf{c}_A^{w+1}$ aient la même longueur que le polygone de contrôle $\mathbf{c}_R^{w+1}$ à l'échelle $w + 1$. Cette étape sera détaillée dans la section 5.2.3. La modification des détails à l'échelle w revient à modifier les points de contrôle d'indices pairs à l'échelle



(a) $\mathbf{c}_R(t)$ et $\mathbf{c}_R^e$ (b) $\mathbf{c}_D(t)$ et $\mathbf{c}_D^e$ (c) $\mathbf{c}_D(t)$ et $\mathbf{c}_D^{w+1}$



(d) $\mathbf{c}_A(t)$ et $\mathbf{c}_A^{w+1}$ (e) $\mathbf{c}_F(t)$ et $\mathbf{c}_F^{w+1}$

Fig. 5.3 – Chaque étape de la boucle d'édition.
De (a) à (e), toutes les étapes de la boucle d'édition sont illustrées. Les courbes fines $\mathbf{c}(t)$ sont représentées en trait plein quand elles sont calculées au cours du processus, en pointillés sinon. Les polygones de contrôle sont représentés au niveau e ou $w+1$ selon les cas.

$w+1$ (en sombre Fig. 5.3(c)). Le résultat est observable Fig. 5.3(d). Le choix d'un niveau intermédiaire $w > e$ augmente le nombre de points de contrôle qui définissent la même portion de courbe. Plus w se rapproche de n, plus ce nombre est grand, et plus la fréquence des plis sera grande (si la courbe est comprimée). Si la courbe est étirée, w se comporte comme un paramètre de raideur.

Reconstruction : La courbe attractive est complètement reconstruite par réinsertion des détails à tous les niveaux $w+1, \ldots, n$. Elle est alors proche de satisfaire la contrainte de longueur (courbe fine Fig. 5.3(d)). L'utilisation du schéma linéaire interpolant risque d'avoir créé des singularités dans la courbe reconstruite, justifiant une étape supplémentaire de lissage.

Conservation de longueur par lissage : Une étape d'optimisation, détaillée en 5.2.4, est appliquée à C_A pour obtenir la courbe finale C_F (Fig. 5.3(e)). Cette courbe satisfait exactement aux contraintes de longueur, et est plus lisse que C_A tout en restant proche, dans un sens que nous préciserons.

5.2.3 Étape une : méthode explicite pour créer une courbe attractive

Objectifs.

Nous détaillons ici la première étape de conservation de la longueur. À partir de la courbe déformée C_D, cette étape calcule en temps linéaire une courbe C_A qui atteint trois objectifs :

- C_A est proche de C_D, c'est-à-dire qu'elle respecte le plus possible la déformation par l'utilisateur,
- elle approxime la contrainte de longueur, ce qui en fait une bonne courbe attractive pour l'étape d'optimisation (section 5.2.4),
- elle crée, si besoin, des plis à une échelle contrôlable.

Replaçons cette étape dans la boucle d'édition (FIG. 5.2). L'utilisateur a édité la courbe initiale C_R par la méthode multirésolution classique, en déplaçant un point de contrôle au niveau e, créant ainsi C_D qui diffère de C_R sur une partie plus ou moins grande (l'étendue dépendant de e). C_D n'a aucune raison de vérifier les contraintes de longueur. Or, pour assurer ces contraintes dans le cas où la courbe est pincée, on s'attend à ce qu'elle forme des plis à une échelle w qui dépend de l'objet physique représenté.

Modification des coefficients d'ondelettes à une échelle intermédiaire.

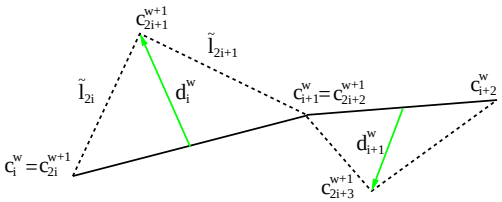


FIG. 5.4 – Étape une : conservation explicite de la longueur.

L'idée que nous allons suivre est d'utiliser les coefficients de détail $\mathbf{d}_i^w$, car ils encodent justement l'information géométrique à cette échelle. Tous les autres coefficients de détail (niveaux $w+1$ à $n-1$) restent intacts. Autant que faire se peut, les coefficients d'échelle $\mathbf{c}_i^w$ sont conservés aussi, ce qui assure la proximité avec C_D. Regardons pour cela la figure 5.4. Les points de contrôle $\mathbf{c}_i^w$ et $\mathbf{c}_{i+1}^w$ viennent d'être modifiés par effet de recomposition suite à l'édition. Nous allons alors calculer un nouveau détail $\mathbf{d}_i^w$ de façon à ce que les longueurs $\tilde{l}_{2i}$ et $\tilde{l}_{2i+1}$ soient égales aux longueurs de référence (longueur des mêmes arêtes dans C_R). Si $\mathbf{c}_i^w$ et $\mathbf{c}_{i+1}^w$ ont été rapprochés, la norme de $\mathbf{d}_i^w$ va augmenter, faisant apparaître ou croître un pli à l'échelle w.

L'origine du vecteur $\mathbf{d}_i^w$ est connue, c'est le milieu du segment $[\mathbf{c}_i^w, \mathbf{c}_{i+1}^w]$ (voir (5.2)). L'extrémité est à une distance $\tilde{l}_{2i}$ de $\mathbf{c}_i^w$, et $\tilde{l}_{2i+1}$ de $\mathbf{c}_{i+1}^w$. Elle appartient donc à l'intersection des deux sphères de centres $\mathbf{c}_i^w$ et $\mathbf{c}_{i+1}^w$, et de rayons $\tilde{l}_{2i}$ et $\tilde{l}_{2i+1}$. Dans le cas général, cette intersection est un cercle, ce qui laisse une infinité de solutions. Afin que la modification entre C_D et C_A soit la plus petite possible, nous choisissons l'unique point du cercle qui appartient au demi-plan délimité par $(\mathbf{c}_i^w, \mathbf{c}_{i+1}^w)$ contenant l'extrémité de l'ancien vecteur $\mathbf{d}_i^w$. Autrement dit, le plan contenant $\{\mathbf{c}_{2i}^{w+1}, \mathbf{c}_{2i+1}^{w+1}, \mathbf{c}_{2i+2}^{w+1}\}$ (voir figure 5.4) est invariant, et $\mathbf{c}_{2i+1}^{w+1}$ reste du même côté de $(\mathbf{c}_i^w, \mathbf{c}_{i+1}^w)$. Nous détaillons juste après comment calculer ce point de façon explicite, ce pour quoi cette étape s'appelle *étape explicite*.

Il nous faut encore prendre en compte les cas où les deux sphères ne s'intersectent pas :
- une sphère est incluse dans l'autre : les deux longueurs sont trop déséquilibrées. Dans ce cas on égalise $\tilde{l}_{2i}$ et $\tilde{l}_{2i+1}$ en conservant leur somme, de façon à préserver la longueur totale.
- les deux sphères sont disjointes : les deux centres $\mathbf{c}_i^w$ et $\mathbf{c}_{i+1}^w$ sont trop écartés. Dans ce cas on rapproche $\mathbf{c}_{i+1}^w$ sur le segment jusqu'à ce que les deux sphères soient tangentes.

Cette procédure doit être appliquée sur chaque segment du polygone $\mathbf{c}^w$, mais pas dans n'importe quel ordre, puisque les sommets $\mathbf{c}_i^w$ sont susceptibles de bouger. Nous partons donc du sommet édité (à l'échelle e, mais qui correspond à un sommet à l'échelle w puisque le schéma est interpolant) car c'est celui que l'utilisateur ne souhaite pas voir bouger. Ensuite nous travaillons sur les segments d'indices croissants puis décroissants le long de la courbe.

Détermination géométrique des coefficients d'ondelette $\mathbf{d}^w$.

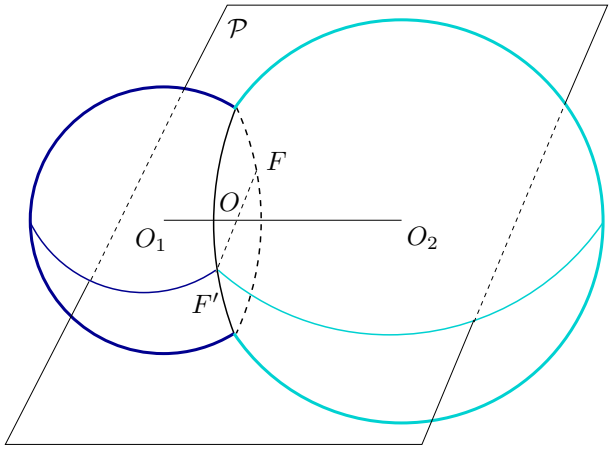


Fig. 5.5 – Intersection de deux sphères.

Notre problème revient à chercher l'intersection de deux sphères et un demi-plan (Fig. 5.5) :

i. la sphère $\mathcal{S}_1(O_1, r_1)$,

ii. la sphère $\mathcal{S}_2(O_2, r_2)$,

iii. le plan affine $\mathcal{P}$ contenant O_1, O_2, et $\frac{1}{2}(O_1 + O_2) + \vec{\delta}$),

instancié avec $O_1 = \mathbf{c}_i^w$, $O_2 = \mathbf{c}_{i+1}^w$, $r_1 = \tilde{l}_{2i}$, $r_2 = \tilde{l}_{2i+1}$ et $\vec{\delta} = \mathbf{d}_i^w$ (détail de la courbe C_D).

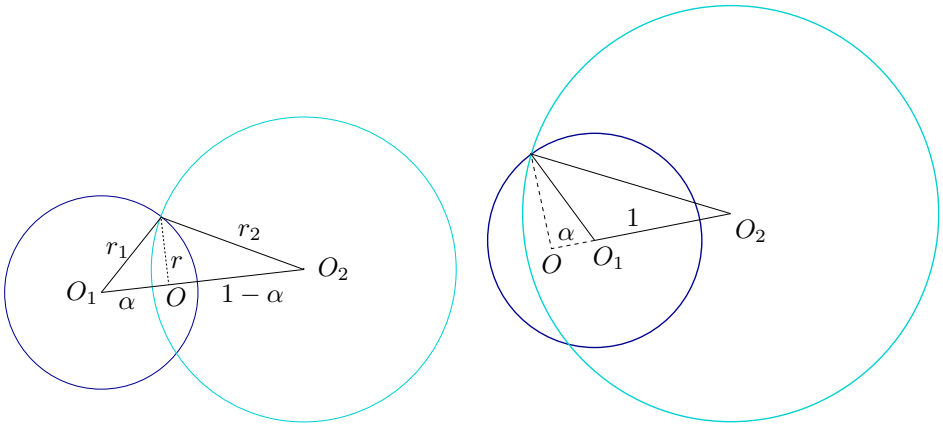
Nous allons d'abord déterminer $\mathcal{S}_1 \cap \mathcal{S}_2$ (un cercle), puis en déduire $\mathcal{S}_1 \cap \mathcal{S}_2 \cap \mathcal{P}$ (deux points), et enfin déterminer dans quel demi-plan est la solution que nous cherchons (choisir parmi les deux points).

Posons $d = dist(O_1, O_2) = \|\overrightarrow{O_1O_2}\|$. Supposons que $\mathcal{S}_1 \cap \mathcal{S}_2 \neq \emptyset$ (*i.e.* $r_1 + r_2 \leq d$ et $|r_1 - r_2| \leq d$), cette intersection est un cercle $\mathcal{C}(O, r)$:

i. de centre $O \in (O_1O_2)$,

ii. de rayon $r \leq \min(r_1, r_2)$,

iii. appartenant à un plan orthogonal à $\overrightarrow{O_1O_2}$ (donc aussi à $\mathcal{P}$).



(a) $\alpha > 0$ (b) $\alpha < 0$

FIG. 5.6 – Calcul du centre et du rayon du cercle d'intersection.

Soient $(1-\alpha, \alpha)$ les coordonnées barycentriques homogènes de O par rapport à $\{O_1, O_2\}$, c'est-à-dire $\overrightarrow{O_1O} = \alpha\ \overrightarrow{O_1O_2}$ (voir FIG. 5.6). En appliquant le théorème de Pythagore dans les triangles (O_1, O, F) et (O_2, O, F), avec $F \in \mathcal{C}$, on déduit r et la position α de O :

$$\left\{ \begin{array}{ll} r^2 + (1-\alpha)^2 d^2 & = r_2^2 \\ r^2 + \alpha^2 d^2 & = r_1^2 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{ll} \alpha & = (d^2 + r_1^2 - r_2^2) \ / \ 2d^2 \\ r^2 & = r_1^2 - \alpha^2 d^2 \end{array} \right.$$

Donc $\mathcal{S}_1 \cap \mathcal{S}_2 \cap \mathcal{P} = \mathcal{C}(O, r) \cap \mathcal{P}$ est l'ensemble de deux points $\{F, F'\}$ tels que :

i. $\overrightarrow{OF}$ et $\overrightarrow{OF'}$ sont orthogonaux à $\overrightarrow{O_1O_2}$,

ii. F et $F' \in \mathcal{P}$, c'est-à-dire $\overrightarrow{OF}$ et $\overrightarrow{OF'}$ sont orthogonaux à $\vec{\delta} \times \overrightarrow{O_1O_2}$,

iii. $\|\overrightarrow{OF}\| = \|\overrightarrow{OF'}\| = r$,

iv. $\overrightarrow{OF} = -\overrightarrow{OF'}$.

On en déduit que $\overrightarrow{OF}$ et $\overrightarrow{OF'}$ sont les deux seuls vecteurs *colinéaires* à $\overrightarrow{O_1O_2} \times (\vec{\delta} \times \overrightarrow{O_1O_2})$, et de norme r. Ces deux vecteurs sont opposés.

Nous connaissons maintenant F et F', parmi lesquels se trouve l'extrémité du vecteur de détail recherché, arbitrairement noté F. On veut que F soit, dans le plan affine $\mathcal{P}$, du même côté de (O_1, O_2) que l'extrémité de $\vec{\delta}$, c'est-à-dire que $\overrightarrow{OF}$ et $\vec{\delta}$ soient dans le même demi-plan vectoriel. Donc $\overrightarrow{OF}$ a le même *sens* que $\overrightarrow{O_1O_2} \times (\vec{\delta} \times \overrightarrow{O_1O_2})$.

Commentaires.

Le principal intérêt de cette méthode est que la courbe résultante C_A est proche de C_D parce que tous leurs coefficients multirésolution sont égaux autant que possible, à part les détails du niveau w. Autrement dit la courbe C_A respecte le plus possible la déformation qu'a appliquée l'utilisateur.

Comme nous le constaterons dans la section 5.2.4, l'étape d'optimisation a besoin, pour bien fonctionner, d'une courbe qui approxime les contraintes de longueur. Or, grâce à la conservation de la longueur des arêtes du polygone au niveau $w+1$, la courbe C_A (une fois reconstruite au

niveau le plus fin) a presque la même longueur que la courbe de référence. C'est donc un bon point de départ pour la deuxième étape, qui lisse utilement la courbe, car elle présente souvent des points anguleux.

5.2.4 Étape deux : optimisation sous contrainte pour corriger la longueur

Nous détaillons ici la deuxième étape du processus qui consiste à corriger la longueur de façon exacte par une optimisation sous contrainte. Nous présentons d'abord les objectifs, puis les calculs, et enfin nous expliquons la complémentarité avec l'étape une.

Partant de C_A, qui ne satisfait qu'approximativement les contraintes de longueur mais possède le squelette des plis, nous devons trouver une courbe finale C_F :

i. dont la longueur est égale à celle de C_R ;

ii. qui est proche de C_A ;

iii. qui est plus lisse que C_A.

La longueur (i) est une contrainte forte qui doit être satisfaite exactement, alors que la proximité (ii) et la régularité (iii) sont des objectifs, des critères d'optimalité. Par conséquence nous allons utiliser une méthode d'optimisation sous contrainte de longueur. La fonction objectif contient un terme de lissage (pour satisfaire (iii)) et un terme de distance qui minimise la distance à la courbe attractive (pour satisfaire (ii)). Les contraintes sont définies par (5.5) pour satisfaire (i).

Énergie de tension comme critère de lissage.

En design variationnel, la notion de courbe ou de surface "lisse", au sens visuel et non analytique, est décrite par un modèle physique [NR83, WW92]. Le critère plus utilisé provient de l'observation de membranes élastiques fines : elles minimisent l'énergie de tension et ont une forme visuellement agréable. L'énergie de tension d'une courbe paramétrique est définie à partir de la courbure κ :

$$\mathcal{E}_B = \int \kappa^2(t)dt$$

Cette expression n'est pas évidente à manipuler parce qu'elle est non linéaire. Suivant une méthode classique [FRSW87, BH94] nous allons utiliser une version linéarisée :

$$\mathcal{E}_B = \int |\mathbf{c}"(t)|^2 dt = \int x"(t)^2 + y"(t)^2 + z"(t)^2 dt \ .$$

Cette approximation est égale à l'énergie exacte dans le cas d'une paramétrisation normale $|\mathbf{c}'(t)| \equiv 1$. Cette formule doit encore être adaptée aux courbes linéaires par morceaux que nous manipulons. Nous utilisons une approximation par différences finies :

$$\mathcal{E}(X,Y,Z) = \sum_{i=1}^{2^n-1} \|\frac{1}{4}(\mathbf{c}_{i-1}^n - 2\mathbf{c}_i^n + \mathbf{c}_{i+1}^n)\|^2 = \frac{1}{2}(X^T H X + Y^T H Y + Z^T H Z),$$

où H est la matrice bande :

$$H = \frac{1}{8}\begin{bmatrix} 1 & -2 & 1 & & & & \\ -2 & 5 & -4 & 1 & & 0 & \\ 1 & -4 & 6 & -4 & 1 & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & 1 & -4 & 6 & -4 & 1 \\ & 0 & & 1 & -4 & 5 & -2 \\ & & & & 1 & -2 & 1 \end{bmatrix},$$

et X, Y, Z sont les composantes de C^n. Nous obtenons une expression quadratique. C'est donc une formulation idéale pour un problème de minimisation.

Recherche de la courbe finale par la résolution d'un problème de minimisation sous contraintes.

Nous nous sommes fixés deux critères d'optimisation. D'une part un critère de lissage (l'énergie de tension ci dessus), et d'autre part un critère de distance, qui doit assurer la proximité entre C_A et C_F. Pour son efficacité bien connue en optimisation, nous choisissons la distance quadratique :

$$\mathcal{D}(X, Y, Z) = \|X - X_A\|^2 + \|Y - Y_A\|^2 + \|Z - Z_A\|^2 .$$

Nous avons maintenant en main les deux termes (quadratiques) que nous équilibrons dans la fonction objectif

$$(1 - \beta)\mathcal{E}(X, Y, Z) + \beta\mathcal{D}(X, Y, Z)$$

à l'aide d'un paramètre $\beta \in [0, 1]$. Des énergies similaires ont déjà été utilisées dans [HHB93]. β est un paramètre de contrôle qui permet de choisir un résultat plus lisse (pour β petit, $\mathcal{E}$ est prépondérant) ou plus proche de la courbe attractive (pour β proche de 1, $\mathcal{D}$ est prépondérant). Notons que nous avons utilisé cette même fonction objectif en section 2.4, et que nous avions conclu que l'énergie de tension était peu utile car les B-splines de degré 2 ou 3 assuraient une régularité acceptable. Il n'en va plus de même ici : les ondelettes B-splines linéaire sont seulement continues. Elles n'assurent donc pas un aspect lisse, et $\mathcal{E}$ va dès lors jouer un rôle important.

En intégrant enfin les contraintes de longueur f_i (définies par (5.5) page 113), notre problème peut se formuler comme une optimisation sous contrainte :

$$\begin{aligned}(X_F, Y_F, Z_F) = \quad & \arg\min \{ (1 - \beta)\mathcal{E}(X, Y, Z) + \beta\mathcal{D}(X, Y, Z) \} \\ & \text{sous contrainte } f_i(X, Y, Z) = 0 \ , \ i = 0, \ldots, N - 2 \ .\end{aligned} \tag{5.6}$$

En utilisant la technique des multiplicateurs de Lagrange [GW73, Cia88], la résolution revient à trouver le (ou les) point(s) stationnaire(s) de la fonction

$$g(X, Y, Z, \Lambda) = (1 - \beta)\mathcal{E} + \beta\mathcal{D} + \sum_{i=0}^{N-2} \lambda_i f_i$$

où $\Lambda = (\lambda_0, \ldots, \lambda_{N-2})^T$ est le vecteur des multiplicateurs de Lagrange. Les contraintes f_i étant quadratiques, g est cubique. Trouver les points stationnaires n'est donc pas aussi immédiat que dans le cas où g est quadratique. Sachant que cette fonction peut avoir plusieurs centaines, voire plusieurs milliers de variables, et se rappelant qu'un de nos objectifs est le temps réel,

nous allons "rendre" g quadratique en linéarisant les f_i. Pour cela nous les remplaçons par leur développement limité $\tilde{f}_i$ au premier ordre par rapport à Δx_i, Δy_i et Δz_i au voisinage de Δx_i^A, Δy_i^A et Δz_i^A :

$$f_i = \tilde{f}_i + O\big((\Delta x_i - \Delta x_i^A)^2 + (\Delta y_i - \Delta y_i^A)^2 + (\Delta z_i - \Delta z_i^A)^2\big)$$

Pour que le développement limité soit effectivement une approximation, il faut que Δx_i, Δy_i etΔz_i (dans la solution) soient proches de Δx_i^A, Δy_i^A etΔz_i^A. Il faut donc que C_A satisfasse presque les contraintes de longueur. Or la courbe fournie par l'étape 1 (section 5.2.3) a justement cette propriété. En plus de contrôler la déformation (à travers le paramètre w), l'étape 1 permet donc à l'étape 2 de bien fonctionner.

Notre problème devient

$$\vec{\nabla}\tilde{g} = \vec{0} \quad \text{où} \quad \tilde{g}(X, Y, Z, \Lambda) = (1-\beta)\mathcal{E} + \beta\mathcal{D} + \sum \lambda_i \tilde{f}_i \ , \tag{5.7}$$

avec

$$\tilde{f}_i = 2\Delta x_i^A \Delta x_i - (\Delta x_i^A)^2 + 2\Delta y_i^A \Delta y_i - (\Delta y_i^A)^2 + 2\Delta z_i^A \Delta z_i - (\Delta z_i^A)^2 - l_i^2.$$

C'est-à-dire un système linéaire en X, Y, Z et Λ :

$$\begin{cases} (1-\beta)HX + 2\beta X + 2\Delta_X \Lambda = 2\beta X_A \\ (1-\beta)HY + 2\beta Y + 2\Delta_Y \Lambda = 2\beta Y_A \\ (1-\beta)HZ + 2\beta Z + 2\Delta_Z \Lambda = 2\beta Z_A \\ 2\Delta_X^T X + 2\Delta_Y^T Y + 2\Delta_Z^T Z = b \end{cases} \tag{5.8}$$

où $b_i = l_i^2 + (\Delta x_i^A)^2 + (\Delta y_i^A)^2 + (\Delta z_i^A)^2$ pour $i = 0 \dots N-2$. Les matrices bandes Δ_X, Δ_Y et Δ_Z, de taille $N \times (N-1)$, sont définies sur le modèle :

$$\Delta_X = \begin{bmatrix} -\Delta x_0^A & & & & \\ \Delta x_0^A & -\Delta x_1^A & & 0 & \\ & \Delta x_1^A & \ddots & & \\ & & \ddots & \ddots & \\ & 0 & & \ddots & -\Delta x_{N-2}^A \\ & & & & \Delta x_{N-2}^A \end{bmatrix}.$$

Résolution du système.

Au vu des matrices H et Δ qui entrent en jeu, ce système est très creux : le nombre de coefficients non nuls du système est en $O(N)$ par rapport au nombre total N^2. Une méthode de résolution itérative semble donc appropriée. Notons de plus que le système est symétrique, mais pas défini positif. Nous avons donc choisi la méthode du gradient bi-conjugué [PVTF02].

Itération de la résolution pour augmenter la précision.

La solution du système (5.8) n'est pas la solution exacte à la minimisation (5.6). En particulier les contraintes de longueur ne sont pas satisfaites exactement. À défaut, afin d'obtenir une solution aussi proche que l'on veut de la solution exacte, nous itérons cette résolution de système. À chaque itération le résultat précédent est utilisé comme nouvelle approximation pour

le développement limité. Autrement dit, la partie du système découlant des $\tilde{f}_i$ (*i.e.* Δ_X, Δ_Y, Δ_Z et b) est recalculée.

Nous n'avons pas de preuve formelle de la convergence de cette itération. Le cas échéant une telle preuve impliquerait la spécification des erreurs du développement limité, donc en particulier une caractérisation précise du point de départ C_A. Néanmoins les résultats sont empiriquement concluants. Dans tous nos tests, allant jusqu'à $257 = 2^8 + 1$ points, il a fallu au maximum 10 itérations pour une précision relative de 10^{-6} (sur la longueur). Pour mettre en lumière ces chiffres, il faut ajouter que les tests incluent de "grosses" déformations. Pour de "petites" déformations successives, typiquement issues d'un *Drag and Drop* (cf. section 1.4), une seule itération s'avère suffisante.

5.3 Création de plis sur des surfaces

La modélisation physique de tissus mous inélastiques est une tâche ardue car cela impliquerait une conservation d'aire et de longueurs, ainsi que la minimisation d'énergies sur la surface. Un comportement caractéristique de la peau ou des étoffes, par exemple, est de former des plis quand ils sont compressés. Nous montrons ici que la conservation de longueur pour courbes 3D précédemment présentée peut servir à créer des plis sur une surface. Nous tirons profit de l'efficacité algorithmique du modèle sur les courbes pour générer des déformation, avec un contrôle intuitif de l'étendue et de la fréquence des plis.

L'idée principale de cette méthode est d'extraire une courbe sur la surface, de la déformer à longueur constante, puis de la ré-injecter dans la surface. Les plis sont propagés sur un voisinage pré-défini, à l'intérieur duquel les plis sont atténués régulièrement. La propagation est assurée en déplaçant chaque sommet du voisinage en fonction de sa distance à la courbe et des points les plus proches sur la courbe.

Les exemples que nous allons présenter utilisent des maillages triangulaires. Néanmoins, du moment qu'on possède la structure de donnée appropriée, toutes sortes de maillages conviennent : quadrangulaires, faces à nombre de côtés différents, maillages non manifolds.

5.3.1 Extraction d'une courbe sur la surface

La première étape consiste à extraire une courbe définie comme une suite d'arêtes du maillage. Cette courbe est linéaire par morceaux et ses points de contrôle sont des sommets du maillage. Selon l'application et selon les outils à disposition, plusieurs méthodes d'extraction sont envisageables :

- Sélection manuelle des sommets.
- Calcul de plus court chemin entre deux extrémités [Mit00].
- Courbe d'intersection avec un autre objet.
- Extraction de lignes de crête [OBS04].
- Extraction de géodésiques [PTBS01].
- Etc.

Une fois extraite, cette courbe est déformée à longueur constante en utilisant la méthode multirésolution présentée dans la section précédente. On obtient ainsi des vecteurs déplacement pour les sommets du maillage correspondant à des points de contrôle de la courbe. Pour créer une déformation surfacique il faut maintenant propager ces déplacements sur le maillage.

5.3.2 Calcul d'un voisinage de la courbe sur le maillage

Il s'agit en premier lieu de définir un voisinage de la courbe sur le maillage, qui sera la zone d'influence de la courbe, c'est-à-dire la partie du maillage qui sera déformée. Pour cela posons deux définitions :

- La distance topologique entre deux sommets est le nombre d'arêtes formant le plus court chemin sur le maillage entre ces deux sommets.
- La distance topologique entre un sommet et la courbe est le minimum des distances topologiques entre ce sommet et les sommets de la courbe.

À titre d'exemple, sur la figure 5.8, les sommets $\mathbf{v}$ et $\mathbf{v}_1$ sont à une distance 2, qui est aussi la distance de $\mathbf{v}$ à la courbe.

Le voisinage de la courbe est alors défini comme l'ensemble des sommets dont la distance topologique à la courbe est bornée (par un nombre arbitraire l_{max}).

Remarque sur le choix du critère de distance.

Cette distance topologique est choisie pour son efficacité. Dans le cas de maillages très irréguliers, dans le sens où les triangles sont distordus, cette distance peut s'avérer insuffisante, car deux sommets topologiquement proches peuvent être éloignés géométriquement. Il conviendrait alors de prendre en compte la distance géométrique. Mais le cas pathologique inverse existe aussi : si une nappe se replie sur elle même, deux points peuvent être proches géométriquement mais loin topologiquement. Par conséquent, le critère de distance le plus robuste serait probablement une distance géométrique *sur la surface*, c'est-à-dire la longueur du plus court chemin entre un sommet et la courbe. Cette robustesse risque de se payer au prix fort d'un point de vue algorithmique, car il faudrait, pour au moins une partie des sommets du maillage, calculer le plus court chemin entre chaque sommet et la courbe (traversant les faces cette fois ci, pas seulement les arêtes). Par conséquent nous nous sommes restreints à la distance topologique, conscients des limites pour les maillages peu réguliers. Cela ne remet pas en cause la suite de notre algorithme, qui ne présume pas de la façon dont est défini le voisinage.

Aspects algorithmiques.

Comme nous le verrons dans la section 5.3.3, nous avons besoin non seulement des sommets du voisinage, mais pour chacun de ses sommets il nous faut aussi connaître tous ses plus proches voisins sur la courbe (au sens topologique). Ce problème peut être vu comme une recherche de plus court chemin entre chaque sommet du maillage et tous les sommets de la courbe. Cette méthode est beaucoup trop coûteuse car le maillage peut avoir plusieurs dizaines ou centaines de milliers de sommets, alors que nous ne cherchons que les plus proches, ceux à une distance de la courbe inférieure à l_{max}. Nous avons donc développé un algorithme plus adéquat, qui consiste à remplir une liste des sommets du voisinage par propagation à partir des sommets de la courbe. Pour cela nous avons besoin des structures de données suivantes :

- *Attente* est une file d'éléments (l, i, j). Elle contient des triplets en attente d'être traités : un chemin de longueur l relie le sommet i du maillage au sommet j appartenant à la courbe.
- $Voisinage$ est une liste d'éléments $(i, l_i, origine)$ triée et non redondante selon i. Elle contient tous les sommets déjà dans le voisinage : le sommet i est à une distance l_i de la courbe, et tous sommets de la courbe à cette distance (connus jusque là) sont rangés dans la liste de sommets non redondante *origine*.

La liste $Voisinage$ est vide au début de l'algorithme. À la fin elle contient tous les sommets du voisinage. L'algorithme 5.1 est écrit en pseudo-code. Voici quelques explications :

- La première boucle **pour** est l'initialisation : on met en attente de traitement tous les sommets de la courbe, qui sont à une distance de 0 à partir d'eux-mêmes.
- La boucle **tant que** traite les sommets en attente un par un tant qu'il y en a, par valeurs croissantes de distance à la courbe. Le traitement est différencié selon que le sommet est dans la liste $Voisinage$ ou pas.
- Les boucles **pour** k voisin de i **faire** servent à propager le voisinage : tous les voisins de i sont mis en attente avec une distance incrémentée. Ce n'est fait que si la distance maximale n'est pas dépassée, ce qui assure que l'algorithme s'arrête.
- Le **sinon** est le cas où i n'est pas encore répertorié comme sommet du voisinage, donc il est ajouté, et le seul plus proche sommet sur la courbe connu à ce moment est j à une distance l.
- *Attente* est une file (FIFO) remplie par valeurs de l incrémentées, donc elle est toujours triée par ordre croissant de l. Une conséquence directe est que si i est dans le voisinage (test **si** $(i, l_i, origine) \in Voisinage$) alors $l \geq l_i$. Donc soit $l > l_i$, soit $l = l_i$. Dans le premier cas il n'y a rien à faire, parce que l'on connaît déjà un chemin plus court que l pour atteindre le sommet i. Dans le deuxième cas, i est déjà dans le voisinage. Comme on cherche *tous* les plus proches voisins de i sur la courbe, il faut répertorier j comme plus proche voisin (*i.e.* l'insérer dans *origine*) si ce n'est pas déjà le cas.

ALGO. **5.1** Calcul du voisinage.

pour i sur la courbe **faire**
  insérer $(0, i, i)$ dans *Attente*
fin pour

tant que *Attente* $\neq \emptyset$ **faire**
  Extraire la tête (l, i, j) de *Attente*
  si $(i, l_i, origine) \in Voisinage$ **alors**
    si $l = l_i$ et $j \notin origine$ **alors**
      insérer j dans *origine*
      si $l < l_{max}$ **alors**
        pour k voisin de i **faire**
          insérer $(l + 1, k, j)$ en queue de *Attente*
        fin pour
      fin si
    fin si
  sinon
    insérer $(i, l, \{j\})$ dans $Voisinage$
    si $l < l_{max}$ **alors**
      pour k voisin de i **faire**
        insérer $(l + 1, k, j)$ en queue de *Attente*
      fin pour
    fin si
  fin si
fin tant que

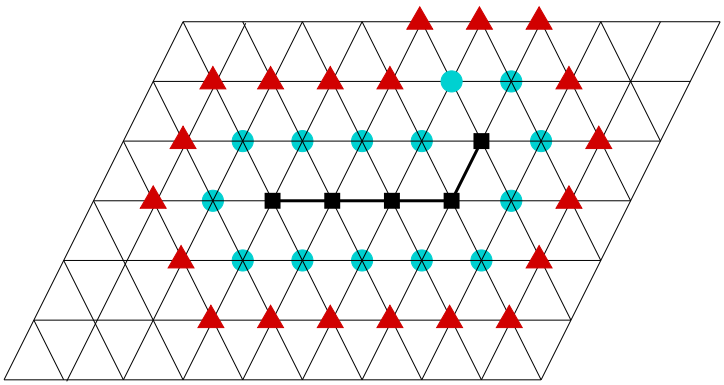


Fig. 5.7 – Calcul du voisinage.
Les sommets de la courbe sont représentés par des carrés,
les 1-voisins par des ronds et les 2-voisins par des triangles.

La complexité de cet algorithme dépend de la régularité du maillage, c'est-à-dire de la valence maximale des sommets. Observons dans un premier temps le cas triangulaire régulier : tous les sommets sont de valence 6. Posons :

- N le nombre de sommets de la courbe (carrés Fig. 5.7),
- n_l le nombre de sommets à une distance l de la courbe, appelés l-voisins (ronds pour $l = 1$ et triangles pour $l = 2$ Fig. 5.7),
- m_l le nombre de plus proches sommets sur la courbe pour un sommet à distance l de la courbe (*i.e.* le nombre d'éléments de la liste *Origine*),
- $C(l)$ le coût de traitement par l'algorithme de tous les points à une distance l de la courbe, c'est-à-dire le coût total de toutes les boucles **tant que** effectuées à distance l.

Pour un maillage triangulaire régulier, en prenant en compte que les sommets de la courbe sont voisins sur le maillage, on peut majorer $n_l \leq 2N + 6l - 2$. Le nombre m_l de plus proches sommets de la courbe est majoré par N et par $6l$ (nombre de l-voisins d'un seul point). En considérant qu'il peut y avoir jusqu'à 6 accès pour un même triplet (l, i, j), le nombre de boucles est majoré par :

$$6n_l m_l \text{ boucles au maximum.}$$

Il reste à calculer le coût maximal de chaque boucle :

- La recherche dans $Voisinage$: linéaire en la longueur de la liste, c'est-à-dire :

$$O\Big(\sum_{k=0}^{l} n_l\Big) = O(lN + l^2).$$

- Les 6 insertions dans *origine* : $6m_l$, petit devant $O(lN + l^2)$.

Par conséquent

$$C(l) = O\big(n_l m_l (lN + l^2)\big) = O\big(l^2(N + l)^2\big).$$

Si l'on prend en compte l'aspect géométrique il est raisonnable de considérer que $l < N$, donc $C(l) = O(l^2 N^2)$. Le coût total de l'algorithme est donc :

$$C = \sum_{l=0}^{l_{max}} C(l) = O\big(l_{max}^3 N^2\big).$$

Ceci nous assure un calcul au plus en quelques secondes pour $N \approx 1000$ sommets et l_{max} de quelques dizaines, ce qui est raisonnable puisqu'il s'agit d'un pré-calcul.

Nous avons fait un petit raccourci dans ce calcul de complexité, dans la mesure où un l-voisin peut se retrouver dans *Attente* avec plusieurs distances différentes, d'où le rôle du test **si** $l = l_i$. Il faut comprendre par là que le nombre de boucles **tant que** effectuées pour une certaine valeur de l peut être en fait supérieur à $6n_l m_l$. Si l'on observe minutieusement l'algorithme, un l-voisin peut se trouver dans *Attente* avec les distances l, $l+1$ et $l+2$ seulement. Donc le surcoût n'est que de l'ordre d'une constante multiplicative, ne modifiant pas le coût asymptotique.

Dans le cas où le maillage n'est pas régulier, ce calcul n'est plus valide. Néanmoins en restant approximatifs, sous l'hypothèse que la valence des sommets est bornée par une valeur petite devant le nombre de sommets, le coût est le même asymptotiquement mais la constante multiplicative change.

5.3.3 Propagation de la déformation

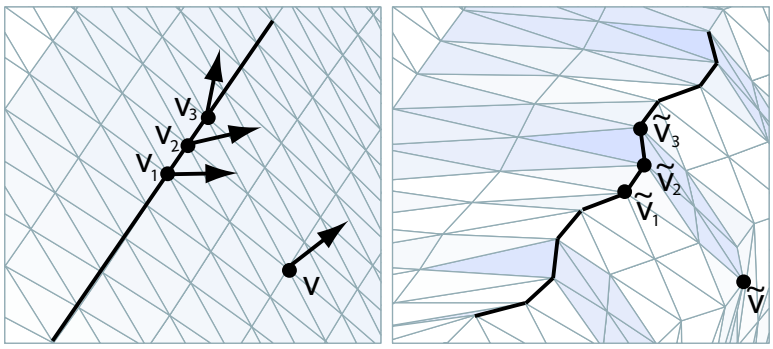


FIG. 5.8 – Propagation de la déformation.
Maillage initial (à gauche) et maillage déformé (à droite). La courbe extraite est en trait épais. Le sommet $\mathbf{v}$ appartient au voisinage. Les sommets $\mathbf{v}_1$, $\mathbf{v}_2$ et $\mathbf{v}_3$ sont ses plus proches voisins sur la courbe. Les vecteurs sont les déplacements de ces sommets.

À ce stade nous connaissons la courbe, un vecteur déplacement pour chacun de ses sommets, et le voisinage. Il reste à propager la déformation, c'est-à-dire définir un déplacement pour chaque sommet de la zone d'influence. Soit $\mathbf{v}$ un sommet de la zone (voir FIG. 5.8). Le sommet correspondant sur la surface déformée est $\tilde{\mathbf{v}} = \mathbf{v} + \vec{\delta}_{\mathbf{v}}$, où $\vec{\delta}_{\mathbf{v}}$ est le vecteur déplacement à définir. Ce sommet peut avoir plusieurs plus proches points sur la courbe par rapport à la distance topologique (ceux stockés dans la liste *origine* précédemment construite). Autrement dit, il existe un ensemble de sommets distincts $\{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ tels que :

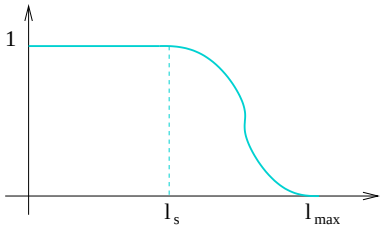
$$dist(\mathbf{v}, \mathbf{v}_1) = \ldots = dist(\mathbf{v}, \mathbf{v}_k) = dist(\mathbf{v}, \text{courbe}) = l_{\mathbf{v}}$$

Nous connaissons les vecteurs déplacement $\vec{\delta}_{\mathbf{v}_1}, \ldots, \vec{\delta}_{\mathbf{v}_k}$ pour chacun de ces points. Il semble logique et réaliste de définir $\vec{\delta}_{\mathbf{v}}$ en fonction de ces déplacements, car cela signifie que $\mathbf{v}$ est influencé par la portion de courbe la plus proche de lui. Nous allons le définir comme une moyenne pondérée des $\vec{\delta}_{\mathbf{v}_i}$. La pondération doit servir à modérer l'impact des distorsions du maillage : deux sommets $\mathbf{v}_i$ et $\mathbf{v}_j$ ont beau être à la même distance topologique $l_{\mathbf{v}}$ de $\mathbf{v}$, les distances géométriques $\|\mathbf{v}-\mathbf{v}_i\|$ et $\|\mathbf{v}-\mathbf{v}_j\|$ sont généralement différentes et l'on voudrait que les sommets aient d'autant plus d'influence qu'ils sont géométriquement proches. Nous choisissons

donc comme poids l'inverse des distances géométriques. En normalisant la somme des poids nous obtenons :

$$\vec{\delta}_{\mathbf{v}} = a(l_{\mathbf{v}}) \left(\sum_{i=1}^{k} \frac{1}{\|\mathbf{v} - \mathbf{v}_i\|} \right)^{-1} \sum_{i=1}^{k} \frac{1}{\|\mathbf{v} - \mathbf{v}_i\|} \vec{\delta}_{\mathbf{v}_i} \ , \tag{5.9}$$

où $a(l)$ est une fonction d'atténuation transversale. Sans elle les plis se propageraient en se mélangeant un peu mais sans perdre de leur amplitude, jusqu'aux bornes de la zone d'influence où ils s'arrêteraient abruptement. $a(l)$ doit décroître de $a(0) = 1$ (*i.e.* quand $\mathbf{v}$ appartient à la courbe) à $a(l_{\mathbf{v}}) = 0$ pour $l_{\mathbf{v}}$ maximum (*i.e.* quand $\mathbf{v}$ est au bord de la zone d'influence). Pour tous nos exemples nous avons choisi $a(l)$ faisant un palier suivi d'un raccord cubique lisse :



$$a(l) = \begin{cases} 1 & \text{si} \quad l < l_s \\ \dfrac{(l - l_{max})^2 (2l - l_s + l_{max})}{(l_s - l_{max})^2 (l_s + l_{max})} & \text{si} \quad l_s < l < l_{max} \end{cases}$$

5.3.4 Résultats

Nous illustrons la création de plis sur un maillage triangulaire par deux exemples (Fig. 5.9 et Fig. 5.10) générés en temps réel. Le calcul de chacune des déformations (calcul du profil et propagation) prend quelques centièmes de seconde sur un Pentium 4 cadencé à 2.6 GHz, en dehors du temps d'extraction et de construction du voisinage.

Fig. 5.9 – Plis à différentes échelles.
Zone d'influence et courbe de profil sur le maillage initial (à gauche). Petits (au milieu) et gros (à droite) plis résultants au dos de la main.

La figure 5.9 illustre le contrôle de la fréquence des plis par le paramètre d'échelle w (voir section 5.2.3). Le maillage initial (à gauche) a 50 000 triangles et la zone d'influence (en bleu/sombre) a 1 400 triangles. La courbe extraite (en rouge) est pincée à longueur constante. Les petits plis (au milieu) sont obtenus avec $w = 3$, et les gros plis (à droite) sont obtenus avec $w = 4$. Le processus de propagation (voir section 5.3) génère automatiquement de larges plis atténués aux extrémités.

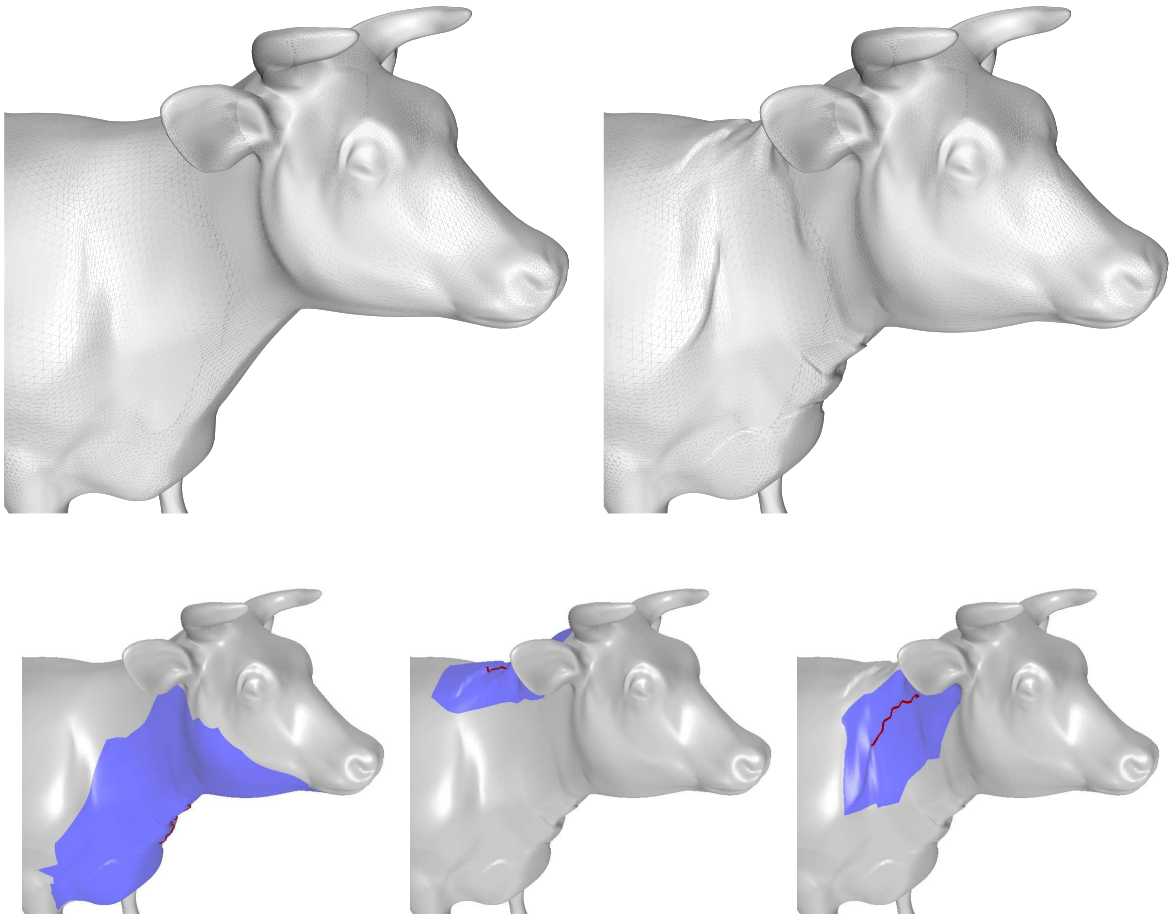


FIG. 5.10 – Superposition de plis.
Le maillage initial (en haut à gauche) subit trois déformations successives (en bas). La surface finale (en haut à droite) présente des plis transversaux autour du cou.

La figure 5.10 montre la superposition de trois déformations (ligne du bas). La surface initiale (en haut à gauche) a 93 000 triangles mais seulement 13 000 triangles ont été modifiés, créant de nombreux plis tout autour du cou, sans modifier la forme globale du maillage. On comprend ici l'intérêt qu'il y aurait à intégrer ce modèle dans un "modeler".

5.3.5 Conclusion

En résumé, nous avons présenté une méthode de déformation à longueur constante de courbes multirésolution 3D linéaires par morceaux. Les courbes sont manipulées via leurs points de contrôle. Bien qu'elle ne s'y réduise pas, cette méthode est principalement dédiée à la création de plis quand deux points sont rapprochés. Il est possible de contrôler l'échelle des plis grâce à la représentation multirésolution. La méthode est ensuite intégrée dans une outil d'édition de maillages pour générer interactivement des plis suite aux déformations.

Le principal atout de ces résultats est qu'ils peuvent s'intégrer dans n'importe quel processus de déformation de surfaces, car ils ne présagent pas de l'origine de ces déformations. Il est possible d'en tirer profit pour animer des surfaces de façon réaliste, comme pour créer des surfaces destinées à être animées indépendamment.

Conclusion

Bilan

Dans ce manuscrit, nous avons étudié l'application de différentes contraintes géométriques non linéaires (aire, volume, longueur) à des modèles de courbes et de surfaces. Tous ces modèles ont pour point commun de permettre une analyse multirésolution, ou pour le moins multi-échelle. Ce sont des outils particulièrement ergonomiques pour manipuler des objets complexes, ce qui explique leur succès en modélisation géométrique, en animation et en visualisation. Grâce à ces modèles, nous pouvons aisément déformer à n'importe quelle échelle des courbes et des surfaces contenant beaucoup de détails, en modifiant quelques points de contrôle. Notre motivation à intégrer des contraintes géométriques dans des modèles multirésolution se comprend comme une volonté de perfectionner les outils d'édition et d'animation de courbes et de surfaces.

Comme l'illustrent nos travaux, dans les situations où l'aspect visuel prime sur le réalisme physique, les contraintes géométriques peuvent imiter le comportement d'objets réels. Les intégrer dans un éditeur, c'est autoriser l'utilisateur à concentrer son attention et son savoir faire sur les objectifs et les contraintes extérieures à l'objet qu'il manipule. Nos travaux atteignent l'objectif de gérer les propriétés géométriques internes à l'objet de façon quasi transparente pour l'utilisateur.

En même temps nous avons développé des méthodes indépendantes du type de déformation appliquée à l'objet. Ces méthodes ont en effet vocation à sortir des éditeurs de courbes et surfaces *stricto sensu*, pour être combinées avec les méthodes actuelles d'animation de scènes complexes, qui utilisent des outils de haut niveau tels que les squelettes ou la détection de collisions.

En outre nous avons balisé ces travaux par des considérations algorithmiques (élaboration d'algorithmes de complexité optimale, mise en place de structures de données appropriées). En implémentant nos méthodes, pour la plupart au sein d'outils d'édition interactifs, nous avons pu valider les résultats théoriques :

Rapidité. Tous nos modèles fonctionnent en temps réel, ce qui est un atout important, si ce n'est un pré-requis, pour être utilisables en animation.

Stabilité. Étant basés au maximum sur des méthodes numériques robustes, ils se sont avérés globalement stables. Nous devons cependant rappeler les quelques difficultés, mentionnées au chapitre 3, rencontrées pour le calcul du volume des surfaces B-splines produit tensoriel lorsque le degré augmente.

Occupation mémoire. La place mémoire occupée par les structures de données, principalement dans le cas des surfaces, était une de nos inquiétudes, ce qui explique nos efforts dans cette direction. Nous l'avons rendue acceptable pour des objets de taille moyenne (plusieurs milliers de points de contrôle). En l'état actuel, l'occupation mémoire reste néanmoins le facteur limitant pour accéder à des surfaces de grande dimension. Ce n'est cependant qu'un critère secondaire, dans la mesure où les données de grande taille sont plutôt l'apanage de

la visualisation. Dans le domaine de l'animation, le souci est plutôt d'obtenir un résultat visuel de bonne qualité à partir de données réduites. Pour cela la multirésolution est un outil pertinent, notamment grâce aux modèles adaptatifs.

objets géométriques	contrainte	chapitre	schéma	espace de param.	synthèse	autres contraintes
courbes B-splines uniformes	aire	2	MR	tore de $\mathbb{R}$	non	oui
surfaces B-splines non uniformes	volume	3.2	ME	pavé de $\mathbb{R}^2$	oui	oui
surfaces B-splines uniformes	volume	3.3	MR	pavé de $\mathbb{R}^2$	non	oui
surfaces triangulaires	volume	4	MR	maillage semi-rég.	non	oui
courbes linéaires par morceaux	longueur	5	MR	intervalle de $\mathbb{R}$	oui	non

TAB. 1 – Comparaisons des méthodes.
Chaque ligne correspond à un modèle. De gauche à droite sont répertoriés : le type d'objets géométriques manipulés, la contrainte appliquée, le chapitre dans lequel est présentée la méthode, le type de schéma (multirésolution (MR) ou multi-échelle (ME)), l'espace de paramétrisation des objets, s'il est nécessaire de reconstruire l'objet dans la base fine à chaque déformation, et la possibilité de combiner d'autres contraintes (linéaires).

Au moment de conclure ces travaux, nous souhaitons revenir sur un certain nombre d'éléments qui permettent de les comparer entre eux. Le tableau 1 regroupe les principales caractéristiques de nos méthodes, et des algorithmes qui les mettent en œuvre.

Penchons nous en premier lieu sur les espaces de paramétrisation. Ceux-ci contraignent le type de schémas multirésolution qu'il est possible de construire. Les courbes paramétrées sur le tore acceptent des schémas multirésolution dits uniformes (*i.e.* les filtres ne dépendent pas de la position), alors que les courbes paramétrées sur un intervalle nécessitent un traitement particulier des extrémités. En étant paramétrées sur des pavés de $\mathbb{R}^2$, les surfaces B-spline produit tensoriel nécessitent également un traitement particulier des bords des patchs. Concernant ces surfaces, l'uniformité (des séquences de nœuds) entre également en ligne de compte. À ce propos la quatrième colonne du tableau rappelle que l'utilisation de séquences non uniformes, en contrepartie d'être plus générales, induit des schémas multi-échelle au lieu de multirésolution. Nous avons établi cette nuance dans les chapitres 2 et 3 : les schémas multi-échelle sont à sens unique, en permettant la subdivision mais pas l'analyse. Enfin, les surfaces paramétrées sur un maillage triangulaire ont l'avantage d'englober des surfaces de topologie quelconque. Leur manipulation nécessite d'utiliser des masques locaux qui définissent les filtres d'analyse et de synthèse.

Un deuxième point important est la nécessité, ou pas, de synthétiser l'objet dans la base fine à chaque déformation. L'avant-dernière colonne du tableau fait écho à la notion de boucle d'édition (FIG. 1.5 page 10) que nous avons introduite dans la section 1.4. Si la contrainte peut être calculée et appliquée dans la base multirésolution, alors tout le processus s'effectue sans aller-retour entre les niveaux de résolution. Ceci permet d'envisager de sortir du domaine de l'animation, pour aller explorer des domaines comme celui de la compression (à ce sujet nous renvoyons le lecteur vers la discussion dans la section 2.5.2).

Nous remarquons que, à l'exception de la contrainte de longueur, tous nos modèles peuvent combiner des contraintes supplémentaires (contraintes linéaires classiques telles que la position ou la tangence). Ceci est un avantage non négligeable pour l'intégration dans un outil d'édition existant. Par contre, dans les cas des surfaces (voir les sections 3.4 et 4.4), nous ne pouvons plus garantir une édition en temps réel, car la résolution des systèmes linéaires est plus complexe.

Cette dernière remarque stigmatise la contrainte de longueur, seule à ne pouvoir être combinée facilement avec d'autres contraintes. Sous plusieurs aspects, celle-ci est particulièrement difficile à traiter, car son expression n'est pas multi-linéaire comme les autres.

Pour finir, citons la possibilité de localiser l'application de la contrainte. Chaque modèle permet de sélectionner une aire d'influence, définissant une partie mobile de la courbe ou de la surface. Ceci augmente la modularité de nos outils, et diminue les temps de calcul. En outre, les schémas multirésolution (par opposition aux schémas multi-échelle) offrent la possibilité de fixer ou laisser libres les coefficients de détail, afin de mieux contrôler l'apparence de la déformation.

Perspectives

Au terme de ce bilan, de multiples extensions semblent possibles à nos travaux. L'une d'elles concerne le calcul du volume englobé par les surfaces triangulaires (chapitre 4). Nous avons proposé de conserver le volume du maillage de contrôle fin au lieu de celui de la surface lisse. Dans la mesure où celle-ci est une surface Box-spline, nous conjecturons qu'il est possible d'en exprimer le volume sous forme tri-linéaire. Pour cela nous envisageons d'utiliser une paramétrisation locale pour calculer une intégrale sur la surface [GOMP98]. L'intégration pourrait se faire de façon directe [LDW97], à l'aide de séries géométriques [HKD93], ou par l'utilisation des relations à deux échelles [DM93, JBU01]. Une autre piste serait d'effectuer une intégration numérique après l'application du schéma de subdivision.

Une autre extension qui semble prometteuse est l'utilisation de filtres modifiés pour prendre en compte des angles aigus et des arêtes vives. Par exemple le schéma de subdivision de Loop a été adapté pour pouvoir conserver ce type de caractéristiques [HDD+94]. La construction, à l'aide du lifting scheme, de schémas multirésolution basés sur ces schémas modifiés [LQS04] pourrait élargir le type des surfaces que l'on peut traiter.

L'utilisation de bases hiérarchiques nous apparaît aussi comme utile et potentiellement fructueuse. De tels modèles ont fait leurs preuves, tant du côté de la modélisation géométrique [FB88, GC95, YHB05] que de la visualisation [BG98]. L'idée générale des modèles hiérarchiques est de raffiner la surface *localement*, là où la géométrie le requiert. Cela permet de diminuer la taille des données traitées pour un résultat similaire, ce qui ne manque pas de nous interpeller, au vu de nos précédentes considérations sur la place mémoire.

Par ailleurs, évoquons l'existence de schémas multirésolution pour les B-splines non uniformes [LMQ01, Ber05]. S'ils se révèlent utilisables avec les contraintes que nous proposons, ils permettraient de dépasser les limites des schémas multi-échelle.

Pour terminer, nous souhaitons élargir le champ d'application. Dans nos travaux, une tâche particulièrement délicate a été d'exprimer la contrainte en base multirésolution, puis de développer un algorithme de calcul viable. Comme nous l'avons déjà souligné, cette étape est un préalable indépendant de l'intégration au processus de déformation. Cela nous intéresse donc de transposer les méthodes de calcul dans des domaines tels que la compression de maillages.

Annexe A

Formulaire B-spline

A.1 B-splines non uniformes

Cette section regroupe un certain nombre de formules classiques sur les fonctions B-splines d'une variables. Les démonstrations se trouvent dans [Far01]. La plupart de ces formules s'appliquent sous forme produit tensoriel et sont utilisées dans le chapitre 3.

Soit $\{N_i^d(t)\}_{0\leq i<m}$ une famille de fonctions B-splines de degré d définies sur une séquence de nœuds $\mathbf{t}\{t_0, t_1, \ldots, t_{m+d}\}$. Une courbe $\mathbf{c}(t)$ s'écrit

$$\mathbf{c}(t) = \sum_{i=0}^{m-1} \mathbf{c}_i N_i^d(t)$$

où $\mathbf{c}_i$ sont les coefficients B-splines, ou points de contrôle.

La **formule des dérivées** permet d'exprimer la dérivée d'une fonction de base de degré d à partir des fonctions de degré $d-1$:

$$\frac{\mathrm{d}}{\mathrm{d}t} N_i^d(t) = \frac{d}{u_{i+d} - u_i} N_i^{d-1}(t) - \frac{d}{u_{i+d+1} - u_{i+1}} N_{i+1}^{d-1}(t)$$

La **récurrence de Boehm** exprime, lorsqu'un nœud $\hat{t}$ est inséré entre t_k et t_{k+1}, les anciennes B-splines en fonction des nouvelles $\hat{N}_i^d$ (définies sur la séquence $\{t_0, \ldots, t_k, \hat{t}, t_{k+1}, \ldots, t_{m+d}\}$) :

$$N_i^d(t) = \frac{\hat{t} - t_i}{t_{i+d} - t_i} \widehat{N}_i^d(t) + \frac{t_{i+d+1} - \hat{t}}{t_{i+d+1} - t_{i+1}} \widehat{N}_{i+1}^d(t) \quad \text{pour} \quad k - d \leq i \leq k$$

L'**algotihme de De Boor** permet de calculer la valeur $\mathbf{c}(t)$ de la courbe un un point t entre t_k et t_{k+1}. En posant la récurrence

$$\mathbf{c}_i^0 = \mathbf{c}_i \quad \text{pour tout } i$$

et pour $1 \leq j \leq d$

$$\mathbf{c}_i^j = \frac{t_{i+d-j+1} - t}{t_{i+d-j+1} - t_i} \mathbf{c}_{i-1}^{j-1} + \frac{t - t_i}{t_{i+d-j+1} - t_i} \mathbf{c}_i^{j-1} \quad \text{pour} \quad i \in \{k-d+j, \ldots, k\} \ ,$$

alors $\mathbf{c}(t) = \mathbf{c}_k^d$.

La **formule d'insertion de nœud** permet, lorsqu'un nœud $\hat{t}$ est inséré entre t_k et t_{k+1}, de calculer les coefficients $\hat{\mathbf{c}}_i$ pour $0 \leq i \leq m$, pour exprimer la même courbe dans la base fine $\hat{N}_i^d$:

- $\hat{\mathbf{c}}_i = \mathbf{c}_i$ pour $i \leq k-d$,
- $\hat{\mathbf{c}}_i = \mathbf{c}_{i-1}$ pour $k+1 \leq i \leq m$,
- pour $k-d+1 \leq i \leq k$

$$\hat{\mathbf{c}}_i = \frac{t_{i+d}-t}{t_{i+d}-t_i}\mathbf{c}_{i-1} + \frac{t-t_i}{t_{i+d}-t_i}\mathbf{c}_i$$

A.2 Ondelettes B-splines uniformes périodiques

Nous donnons ici le détail des formules et matrices utiles pour les B-splines uniformes périodiques utilisées dans le chapitre 2.

Courbes B-spline périodiques.

Les fonctions B-splines uniformes sont définies à partir d'une séquence de nœuds $\mathbf{t} = \{t_i\}_{i\in\mathbb{Z}}$ uniforme, *i.e.* il existe un pas h tel que pour tout $i \in \mathbb{Z}$

$$t_i = ih.$$

Les fonctions de base ont la propriété d'être invariantes par translation :

$$N^d_{i+k}(t) = N^d_i(t-kh) \tag{A.1}$$

Pour définir des courbes B-splines périodiques, les espaces emboités V^j sont construits pour être de dimension $p2^j$ pour $0 \leq j \leq n$. L'espace V^n est engendré par les périodisées des fonctions B-spline sur la séquence de nœuds à pas $h = \frac{1}{p2^n}$:

$$\varphi^n_i(t) = \sum_{k\in\mathbb{Z}} N^d_i(t-k) = \sum_{k\in\mathbb{Z}} N^d_{i+kp2^n}(t)$$

Formules multirésolution dans le cas quadratique.

Les matrices de décomposition sont circulantes, construites à partir des équations (2.2) et (2.3) :

$$A^j = \frac{1}{4}\begin{bmatrix} 3 & -1 & 0 & \cdots & & 0 & -1 & 3 \\ -1 & 3 & 3 & -1 & 0 & \cdots & & 0 \\ & \ddots & \ddots & \ddots & \ddots & & & \\ & & -1 & 3 & 3 & -1 & 0 & 0 \\ 0 & & & & -1 & 3 & 3 & -1 \end{bmatrix}$$

$$B^j = \frac{1}{4}\begin{bmatrix} 3 & -1 & 0 & \cdots & & 0 & 1 & -3 \\ 1 & -3 & 3 & -1 & 0 & \cdots & & 0 \\ & \ddots & \ddots & \ddots & \ddots & & & \\ & & 1 & -3 & 3 & -1 & 0 & 0 \\ 0 & & & & 1 & -3 & 3 & -1 \end{bmatrix}$$

et les matrices de reconstruction à partir des équations (2.4) et (2.5) :

$$P^j = \frac{1}{4}\begin{bmatrix} 3 & 1 & 0 & \cdots & 0 \\ 1 & 3 & 0 & \cdots & 0 \\ 0 & 3 & 1 & & \\ 0 & 1 & 3 & & \\ & & \ddots & \ddots & \\ & & & \ddots & 1 \\ & & & & 3 \\ 1 & 0 & \cdots & 0 & 3 \\ 3 & 0 & \cdots & 0 & 1 \end{bmatrix} \qquad Q^j = \frac{1}{4}\begin{bmatrix} 3 & -1 & 0 & \cdots & 0 \\ 1 & -3 & 0 & \cdots & 0 \\ 0 & 3 & -1 & & \\ 0 & 1 & -3 & & \\ & & \ddots & \ddots & \\ & & & \ddots & -1 \\ & & & & -3 \\ -1 & 0 & \cdots & 0 & 3 \\ -3 & 0 & \cdots & 0 & 1 \end{bmatrix}$$

Mesures sur les courbes.

La matrice initiale servant au calcul d'aire est :

$$M^n = \frac{1}{12}\begin{bmatrix} 0 & 10 & 1 & 0 & \cdots & 0 & -1 & -10 \\ -10 & 0 & 10 & 1 & 0 & \cdots & 0 & -1 \\ & \ddots & \ddots & \ddots & & & & \\ & & \ddots & \ddots & \ddots & & 0 & \\ & 0 & & \ddots & \ddots & \ddots & & \\ & & & & \ddots & \ddots & \ddots & \\ 1 & 0 & \cdots & 0 & -1 & -10 & 0 & 10 \\ 10 & 1 & 0 & \cdots & 0 & -1 & -10 & 0 \end{bmatrix}$$

La matrice initiale servant à exprimer l'énergie de tension est :

$$H = \frac{1}{8}\begin{bmatrix} 6 & -4 & 1 & & & & & 1 & -4 \\ -4 & 6 & -4 & 1 & & & & & 1 \\ 1 & -4 & 6 & -4 & 1 & & & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & & 0 & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & & \\ & 0 & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & & & 1 & -4 & 6 & -4 & 1 \\ 1 & & & & & 1 & -4 & 6 & -4 \\ -4 & 1 & & & & & 1 & -4 & 6 \end{bmatrix},$$

A.3 Ondelettes B-splines uniformes sur un intervalle

B-splines unidimensionnelles : les courbes.

Les B-splines définies sur un intervalle sont construites sur une séquence de nœuds $\mathbf{t} = \{t_0, t_1, \ldots, t_{m+d}\}$ finie telle que les nœuds de bord sont de multiplicité d (le degré) :

$$t_0 = t_1 = \ldots = t_d \text{ et } t_m = \ldots = t_{m+d} \ ,$$

et les nœuds intérieurs sont répartis uniformément, c'est-à-dire :

$$t_{d+i} = i\ \frac{t_m - t_d}{m-d} \text{ pour } 0 \leq i \leq m-d\ .$$

Sur cette séquence est construite une base de B-splines $N_i(t)$ de degré d, pour $0 \leq i \leq m-1$. Une courbe $\mathbf{c}(t)$ dans l'espace engendré est :

$$\begin{aligned} \mathbf{c} \quad : \quad [0,1] &\longrightarrow \mathbb{R}^2 \text{ ou } \mathbb{R}^3 \\ t \quad &\longrightarrow \mathbf{c}(t) = \sum_{i=0}^{m-1} \mathbf{c}_i\ N_i(t) \end{aligned}$$

où $\mathbf{c}_i$ sont les points de contrôle dans $\mathbb{R}^2$ ou $\mathbb{R}^3$. L'espace ainsi généré est de dimension m.

Analyse multirésolution unidimensionnelle.

Pour construire une analyse en ondelettes il faut définir les espaces emboîtés et des bases pour ceux-ci. L'espace V^n, de dimension $m_n = d + p2^n$, est l'espace construit ci-dessus avec $m = m_n$. Les fonctions d'échelle φ_i^n qui l'engendrent sont donc les B-splines de degré d :

$$\varphi_i^n(t) = N_i(t)$$

sur la séquence uniforme $\mathbf{t}^n = \mathbf{t}$. Par simplicité nous supposerons que $t_d = 0$ et $t_m = 1$, donc :

$$t_{d+i} = \frac{i}{p2^n} \quad \text{pour } 0 \leq i \leq p2^n\ .$$

Puis les espaces emboités V^j et W^j sont construits pour avoir les dimensions :

$$dim(V^j) = m_j = d + p2^j \quad \text{et} \quad dim(W^j) = m_{j+1} - m_j = p2^j$$

Pour définir les espaces d'approximation $V^0 \subset V^1 \subset \cdots \subset V^n$ nous définissons récursivement des séquences de nœuds $\mathbf{t}^j$ emboîtées $\mathbf{t}^0 \subset \mathbf{t}^1 \subset \cdots \subset \mathbf{t}^n$ par conservation des nœuds de bords

$$t_i^{j-1} = t_i^j = 0 \quad \text{et} \quad t_{m_{j-1}+i}^{j-1} = t_{m_j+i}^j = 1\ , \quad \text{pour } 1 \leq i \leq d\ ,$$

et par supression d'un nœud intérieur sur deux :

$$t_{d+i}^{j-1} = t_{d+2i}^j \quad \text{pour } 1 \leq i \leq p2^{j-1} - 1\ ,$$

ce qui revient à dire

$$t_{d+i}^j = \frac{i}{p2^j} \quad \text{pour} \quad 0 \leq j \leq n \quad \text{et} \quad 0 \leq i \leq p2^j\ .$$

Chaque espace V^j est alors engendré par

$$\varphi_i^j(t) = N_i^j(t) \quad \text{pour } 0 \leq i \leq m_j$$

où N_i^j est la spline d'indice i et de degré d sur la séquence décimée $\mathbf{t}^j$.

Ces choix définissent de manière unique les filtres A^j et P^j (voir section 1.3). Il y a maintenent plusieurs façons de définir les ondelettes ψ^j engendrant les espaces de détails W^j ou, de façon équivalente, de définir les filtres B^j et Q^j. Dans la mesure où notre principal intérêt est d'avoir des filtres de taille réduite, nous avons choisi un schéma biorthogonal [SDS96]. Le paragraphe suivant donne ces filtres pour les B-splines quadratiques.

matrice	A^j	B^j	P^j	Q^j
taille	$m_{j-1}\times m_j =$ $(d+p2^{j-1})\times(d+p2^j)$	$(m_j - m_{j-1})\times m_j$ $= p2^{j-1}\times(d+p2^j)$	$m_j\times m_{j-1} =$ $(d+p2^j)\times(d+p2^{j-1})$	$m_j\times(m_j - m_{j-1})$ $= (d+p2^j)\times p2^{j-1}$

Tab. A.1 – Taille des matrices de décomposition et reconstruction.

Matrices d'analyse et de synthèse pour B-splines quadratiques.

$$A^1 = \frac{1}{4}\begin{bmatrix} 4 & 0 & 0 & 0 \\ -2 & 4 & 4 & -2 \\ 0 & 0 & 0 & 4 \end{bmatrix} \quad \text{et} \quad B^1 = \frac{1}{4}\begin{bmatrix} 2 & -4 & 4 & -2 \end{bmatrix}$$

$$P^1 = \frac{1}{4}\begin{bmatrix} 4 & 0 & 0 \\ 2 & 2 & 0 \\ 0 & 2 & 2 \\ 0 & 0 & 4 \end{bmatrix} \quad \text{et} \quad Q^1 = \frac{1}{4}\begin{bmatrix} 0 \\ -2 \\ 2 \\ 0 \end{bmatrix}$$

Puis pour $j \geq 2$:

$$A^j = \frac{1}{4}\begin{bmatrix} 4 & & & & & & & & & & \\ -2 & 4 & 3 & -1 & & & & & & & \\ & & -1 & 3 & 3 & -1 & & & & & \\ & & & & -1 & 3 & 3 & -1 & & & \\ & & & & & & \ddots & & & & \\ & & & & & & -1 & 3 & 3 & -1 & \\ & & & & & & & & -1 & 3 & 4 & -2 \\ & & & & & & & & & & & 4 \end{bmatrix}$$

$$B^j = \frac{1}{4}\begin{bmatrix} 2 & -4 & 3 & -1 & & & & & & & \\ & & 1 & -3 & 3 & -1 & & & & & \\ & & & & 1 & -3 & 3 & -1 & & & \\ & & & & & & \ddots & & & & \\ & & & & & & 1 & -3 & 3 & -1 & \\ & & & & & & & & 1 & -3 & 4 & -2 \end{bmatrix}$$

$$P^j = \frac{1}{4}\begin{bmatrix} 4 & & & & & & & & \\ 2 & 2 & & & & & & & \\ & 3 & 1 & & & & & & \\ & 1 & 3 & & & & & & \\ & & 3 & 1 & & & & & \\ & & 1 & 3 & & & & & \\ & & & 3 & & & & & \\ & & & 1 & \ddots & & & & \\ & & & & & 1 & & & \\ & & & & & 3 & & & \\ & & & & & 3 & 1 & & \\ & & & & & 1 & 3 & & \\ & & & & & & 2 & 2 & \\ & & & & & & & 4 & \end{bmatrix} \quad \text{et} \quad Q^j = \frac{1}{4}\begin{bmatrix} 0 & & & & & & & & \\ -2 & & & & & & & & \\ 3 & -1 & & & & & & & \\ 1 & -3 & & & & & & & \\ & 3 & -1 & & & & & & \\ & 1 & -3 & & & & & & \\ & & 3 & & & & & & \\ & & 1 & \ddots & & & & & \\ & & & & & -1 & & & \\ & & & & & -3 & & & \\ & & & & & 3 & -1 & & \\ & & & & & 1 & -3 & & \\ & & & & & & 2 & & \\ & & & & & & 0 & & \end{bmatrix}$$

Surfaces produit tensoriel.

Un espace de surfaces V^n est défini comme produit tensoriel $V^n = V^n_{\mathbf{u}} \otimes V^n_{\mathbf{v}}$ de deux espaces $V^n_{\mathbf{u}}$ et $V^n_{\mathbf{v}}$ définies comme précédamment, sur deux séquences de nœuds uniformes

$$\mathbf{u} = \{u_0, u_1, \ldots, u_{m_n+d}\} \quad \text{et} \quad \mathbf{v} = \{v_0, v_1, \ldots, v_{m_n+d}\} \ ,$$

et munis de bases multirésolution :

$$\left(\varphi^0_{\mathbf{u}}, \psi^0_{\mathbf{u}}, \psi^1_{\mathbf{u}}, \ldots, \psi^{n-1}_{\mathbf{u}}\right) \quad \text{base de} \quad V^n_{\mathbf{u}} = V^0_{\mathbf{u}} \bigoplus_{j=0}^{n-1} W^j_{\mathbf{u}}$$

$$\left(\varphi^0_{\mathbf{v}}, \psi^0_{\mathbf{v}}, \psi^1_{\mathbf{v}}, \ldots, \psi^{n-1}_{\mathbf{v}}\right) \quad \text{base de} \quad V^n_{\mathbf{v}} = V^0_{\mathbf{v}} \bigoplus_{j=0}^{n-1} W^j_{\mathbf{v}}$$

Remarque : en toute généralité le degré d et la dimension m_n peuvent être différents selon la dimension u ou v. Pour ne pas allourdir les notations nous supposerons qu'ils sont égaux.

Les fonctions d'échelle bidimensionnelles sont les produits tensoriels des fonctions d'échelle unidimensionnelles :

$$\varphi^n_{\mathbf{i}}(u,v) = \varphi^n_{i_u}(u)\varphi^n_{i_v}(v) \quad \text{pour } (0,0) \leq \mathbf{i} = (i_u, i_v) \leq (m_n - 1, m_n - 1) \ .$$

La décomposition des espaces bidimensionnels est plus complexe car elle résulte du produit tensoriel des décompositions unidimensionnelles. Pour chaque niveau de résolution j, l'espace V^j se décompose avec un espace d'approximation V^j et avec *trois* espaces de détails W^{j-1}_0, W^{j-1}_1 et W^{j-1}_2 sous la forme $V^j = V^{j-1} \oplus W^{j-1}_0 \oplus W^{j-1}_1 \oplus W^{j-1}_2$. Le tableau A.2 récapitule les propriétés de chacun de ces espaces. Les fonctions d'échelle et les ondelettes sont représentées figure A.1. Une surfaces S (ou un patch) de l'espace V^n s'écrit alors dans la base fine :

$$\begin{aligned} S \ : \ [0,1]^2 &\longrightarrow \mathbb{R}^3 \\ (u,v) &\longrightarrow S(u,v) = \sum_{\mathbf{i}=(0,0)}^{(m_n-1,m_n-1)} \mathbf{p_i}\ \varphi^n_{\mathbf{i}}(u,v) \ . \end{aligned}$$

espace	dimension	fonctions de base	coefficients
$V^j = V^j_{\mathbf{u}} \otimes V^j_{\mathbf{v}}$	$m_j \times m_j$	$\varphi^j_{\mathbf{i}}(u,v) = \varphi^j_{i_u}(u)\varphi^j_{i_v}(v)$	$\mathbf{p}^j_{\mathbf{i}}$
$W^j_0 = W^j_{\mathbf{u}} \otimes V^j_{\mathbf{v}}$	$(m_{j+1} - m_j) \times m_j$	$\psi^j_{0,\mathbf{i}}(u,v) = \psi^j_{i_u}(u)\varphi^j_{i_v}(v)$	$\mathbf{d}^j_{0,\mathbf{i}}$
$W^j_1 = V^j_{\mathbf{u}} \otimes W^j_{\mathbf{v}}$	$m_j \times (m_{j+1} - m_j)$	$\psi^j_{1,\mathbf{i}}(u,v) = \varphi^j_{i_u}(u)\psi^j_{i_v}(v)$	$\mathbf{d}^j_{1,\mathbf{i}}$
$W^j_2 = W^j_{\mathbf{u}} \otimes W^j_{\mathbf{v}}$	$(m_{j+1} - m_j) \times (m_{j+1} - m_j)$	$\psi^j_{2,\mathbf{i}}(u,v) = \psi^j_{i_u}(u)\psi^j_{i_v}(v)$	$\mathbf{d}^j_{2,\mathbf{i}}$

Tab. A.2 – Décomposition des espaces multirésolution produit tensoriel.

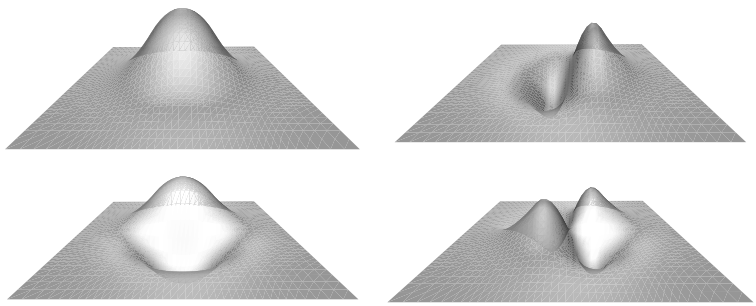


Fig. A.1 – Ondelettes B-spline quadratiques uniformes produit tensoriel.
De gauche à droite et de haut en bas : φ, ψ_0, ψ_1 et ψ_2.

où $\mathbf{p_i} \in \mathbb{R}^3$ sont les coefficients d'échelle, ou points de contrôle de la surface. Dans la base décomposée

$$V^n = V^0 \bigoplus_{j=0}^{n-1} \left(W_0^j \oplus W_1^j \oplus W_2^j \right)$$

elle se reformule :

$$S(u,v) = \sum_{\mathbf{i}=(0,0)}^{(m_0-1,m_0-1)} \mathbf{p}_\mathbf{i}^0 \; \varphi_\mathbf{i}^0(u,v)$$

$$+ \sum_{j=0}^{n-1} \left(\sum_\mathbf{i} \mathbf{d}_{0,\mathbf{i}}^j \; \psi_{0,\mathbf{i}}^j(u,v) \; + \; \sum_\mathbf{i} \mathbf{d}_{1,\mathbf{i}}^j \; \psi_{1,\mathbf{i}}^j(u,v) \; + \; \sum_\mathbf{i} \mathbf{d}_{2,\mathbf{i}}^j \; \psi_{2,\mathbf{i}}^j(u,v) \right)$$

où $\mathbf{d}_{0,\mathbf{i}}^j$, $\mathbf{d}_{1,\mathbf{i}}^j$ et $\mathbf{d}_{2,\mathbf{i}}^j$ sont trois séries de coefficients de détails correspondant respectivement aux espaces W_0^j, W_1^j et W_2^j.

Banc de filtres produit tensoriel.

L'analyse (décomposition) et la synthèse (recomposition) d'une surface dans les différentes bases multirésolution se fait grâce à un banc de filtres similaire à celui présenté pour le cas unidimensionnel (FIG. 1.1), mais où les trois séries de détails sont mis de coté à chaque étape (voir FIG. A.2).

$$\begin{array}{ccccccccc}
(\mathbf{p}^n) & \longrightarrow & (\mathbf{p}^{n-1}) & \longrightarrow & \cdots & & (\mathbf{p}^1) & \longrightarrow & (\mathbf{p}^0) \\
 & \searrow & (\mathbf{d}_0^{n-1}) & \searrow & & & & \searrow & (\mathbf{d}_0^0) \\
 & & (\mathbf{d}_1^{n-1}) & & \cdots & & \cdots & & (\mathbf{d}_1^0) \\
 & & (\mathbf{d}_2^{n-1}) & & & & & & (\mathbf{d}_2^0)
\end{array}$$

FIG. A.2 – Banc de filtres produit tensoriel.

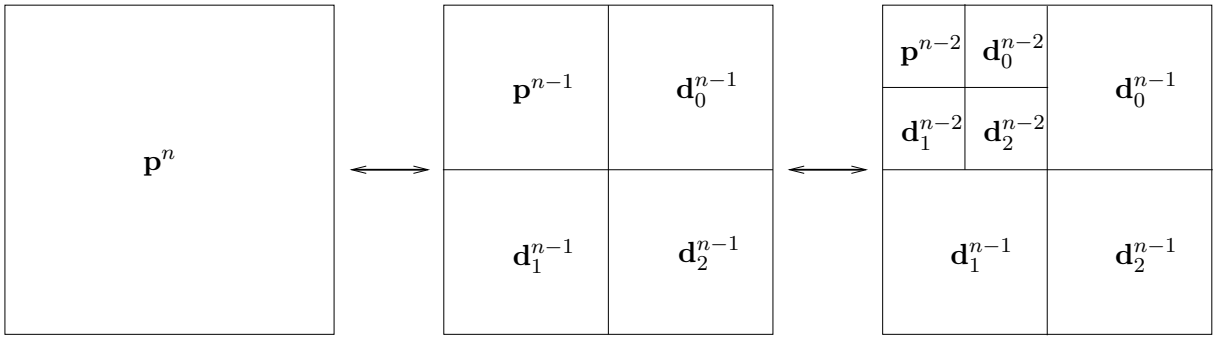


FIG. A.3 – Stockage des coefficients multirésolution.

En considérant l'ensemble des points de contrôle comme une matrice $\mathbf{p}^j = (\mathbf{p}_{i_u,i_v}^j)_{i_u,i_v}$ de points de $\mathbb{R}^3$, et de la même manière les détails $\mathbf{d}_0^j$, $\mathbf{d}_1^j$ et $\mathbf{d}_2^j$, chaque étape d'analyse applique

les filtres sous forme produit tensoriel sur certains blocs de la matrice (voir Fig. A.3) :

$$\begin{aligned}
\mathbf{p}^{j-1} &= A^j\,\mathbf{p}^j\,(A^j)^T \\
\mathbf{d}_0^{j-1} &= B^j\,\mathbf{p}^j\,(A^j)^T \\
\mathbf{d}_1^{j-1} &= A^j\,\mathbf{p}^j\,(B^j)^T \\
\mathbf{d}_2^{j-1} &= B^j\,\mathbf{p}^j\,(B^j)^T
\end{aligned}$$

De la même façon la synthèse s'écrit :

$$\mathbf{p}^j = P^j\,\mathbf{p}^{j-1}\,(P^j)^T \;+\; Q^j\,\mathbf{d}_0^{j-1}\,(P^j)^T \;+\; P^j\,\mathbf{d}_1^{j-1}\,(Q^j)^T \;+\; Q^j\,\mathbf{d}_2^{j-1}\,(Q^j)^T\ .$$

Les équations équivalentes (voir Eq. (1.3) et (1.4) page 5) existent pour les fonctions de base (voir Fig. A.1) :

$$\begin{aligned}
\varphi^{j-1} &= (P^j)^T\,\varphi^j\,P^j \\
\psi_0^{j-1} &= (Q^j)^T\,\varphi^j\,P^j \\
\psi_1^{j-1} &= (P^j)^T\,\varphi^j\,Q^j \\
\psi_2^{j-1} &= (Q^j)^T\,\varphi^j\,Q^j \\
\varphi^j &= (A^j)^T\,\varphi^{j-1}\,A^j \;+\; (B^j)^T\,\psi_0^{j-1}\,A^j \;+\; (A^j)^T\,\psi_1^{j-1}\,B^j \;+\; (B^j)^T\,\psi_2^{j-1}\,B^j
\end{aligned}$$

Bibliographie

[BCR91] G. Beylkin, R. R. Coifman, and V. Rokhlin. Fast wavelet transforms and numerical algorithms I. *Comm. Pure and Appl. Math.*, 44 :141–183, 1991. 5

[Ber04] Martin Bertram. Biorthogonal loop-subdivision wavelets. *Computing*, 72(1-2) :29 – 39, 2004. 82, 92

[Ber05] Martin Bertram. Single-knot wavelets for non-uniform b-splines. *Computer-Aided Geometric Design (CAGD)*, to appear, 2005. 133

[BG98] Georges-Pierre Bonneau and Alexandre Gerussi. Hierarchical decomposition of datasets on irregular surface meshes. In *CGI '98 Proceedings*. IEEE Computer Society, 1998. 133

[BH94] G-P. Bonneau and H. Hagen. Variational design of rational bézier curves and surfaces. *Curves and Surfaces in Geometric Design*, pages 51–58, 1994. 22, 119

[BHN96] Georges-Pierre Bonneau, Stefanie Hahmann, and Gregory M. Nielson. Blac-wavelets : a multiresolution analysis with non-nested spaces. In *VIS '96 : Proceedings of the conference on Visualization*, pages 43–48. IEEE Computer Society Press, 1996. 5

[BK03] Mario Botsch and Leif Kobbelt. Multiresolution surface representation based on displacement volumes. *Computer Graphics Forum*, 22(3) :483–491, 2003. (Proceedings Eurographics '03). 2

[BKMTK00] Laurence Boissieux, Gergo Kiss, Nadia Magnenat-Thalmann, and Prem Kalra. Simulation of skin aging and wrinkles with cosmetics insight. In *Computer animation and simulation*, pages 15–27, 2000. 110

[Bli78] James F. Blinn. Simulation of wrinkled surfaces. In *SIGGRAPH Computer Graphics*, pages 286–292, 1978. 110

[Boe80] Wolfgang Boehm. Inserting new knots into a bspline curve. *Computer-Aided Design*, 12 :199–201, 1980. 43

[Bon98] Georges-Pierre Bonneau. Multiresolution analysis on irregular surface meshes. *IEEE Transactions on Visualization and Computer Graphics*, 4(4) :365–378, 1998. 1

[BW98] David Baraff and Andrew Witkin. Large steps in cloth simulation. *Computer Graphics*, 32(Annual Conference Series) :43–54, 1998. 110

[CG91] George Celniker and Dave Gossard. Deformable curve and surface finite-elements for free-form shape design. *SIGGRAPH Computer Graphics*, pages 257–266, 1991. 2

[Cha74] G. Chaikin. An algorithm for high speed curve generation. *Computer Graphics and Image Processing*, 3 :346–349, 1974. 15

[Cia88] Philippe G. Ciarlet. *Introduction to Numerical Linear Algebra and Optimization*. Cambridge University Press, 1988. 22, 120

[CM01] Albert Cohen and Basarab Matei. Compact representation of images by edge adapted multiscale transforms. In *Proceedings of Image Processing 01*, volume 1, pages 8–11, 2001. 5

[CMPS97] P. Cignoni, C. Montani, E. Puppo, and R. Scopigno. Multiresolution representation and visualization of volume data. *IEEE Transactions on Visualization and Computer Graphics*, 3(4) :352–369, 1997. 1

[CPD+96] Andrew Certain, Jovan Popović, Tony DeRose, Tom Duchamp, David Salesin, and Werner Stuetzle. Interactive multiresolution surface viewing. *SIGGRAPH Computer Graphics*, pages 91–98, 1996. 98

[CQ92] C. Chui and E. Quak. Wavelets on a bounded interval. In D. Braess and L. L. Schumaker, editors, *Numerical Methods of Approximation Theory*, volume 9, pages 53–75. Birkhäuser Verlag, Basel, 1992. 1

[DDCB00] Gilles Debunne, Mathieu Desbrun, Marie-Paule Cani, and Alan H. Barr. Adaptive simulation of soft bodies in real-time. In *Computer Animation 2000, Philadelphia, USA*, pages 133–144, May 2000. 2

[DG95] Mathieu Desbrun and Marie-Paule Gascuel. Animating soft substances with implicit surfaces. In *SIGGRAPH Computer Graphics*, pages 287–290. ACM Press, 1995. 2

[DLG90] Nira Dyn, David Levin, and John A. Gregory. A butterfly subdivision scheme for surface interpolation with tension control. *Transactions on Graphics (TOG)*, 9(2) :160–169, april 1990. 84, 88

[DM93] W. Dahmen and C.A. Micchelli. Using the refinement equation for evaluating integrals of wavelet. *SIAM J. Numer. Anal.*, 30 :507–537, 1993. 133

[DSB99] Mathieu Desbrun, Peter Schröder, and Alan Barr. Interactive animation of structured deformable objects. In *Graphics Interface*, pages 1–8, 1999. 110

[EL91] D. Eberly and J. Lancaster. On gray scale image measurements i. arc length and area. *GMIP*, 53(6) :538–549, 1991. 16, 29

[Elb00] G. Elber. Linearizing the area and volume constraints. *Technical Report CIS-2000-04*, 2000. 2, 14, 16, 22, 27, 80

[Elb01] G. Elber. Multiresolution curve editing with linear constraints. In *6th ACM/IEEE Symposium on Solid Modeling and Applications*, pages 109–119. Ann Arbor, Michigan, June 2001. v, 2, 14, 27, 28, 30, 31, 38, 39, 69, 80

[Far01] Gerald Farin. *Curves and surfaces for CAGD : a practical guide*. Morgan Kaufmann Publishers Inc., 2001. 4, 28, 37, 38, 43, 135

[FB88] David R. Forsey and Richard H. Bartels. Hierarchical b-spline refinement. *SIGGRAPH Computer Graphics*, 22(4) :205–212, 1988. 133

[Fow92] Barry Fowler. Geometric manipulation of tensor product surfaces. In *Proceedings of the 1992 symposium on Interactive 3D graphics*, pages 101–108. ACM Press, 1992. 2

[FRSW87] G. Farin, G. Rein, N. Sapidis, and A.J. Worsey. Fairing cubic b-spline curves. *Computer Aided Geometric Design*, 4 :91–103, 1987. 22, 119

[FS94] Adam Finkelstein and David H. Salesin. Multiresolution curves. *SIGGRAPH Computer Graphics*, pages 261–268, 1994. 1, 5

[GBS99] Laurent Grisoni, Carole Blanc, and Christophe Schlick. Hb-splines : a blend of hermite splines and b-splines. *Computer Graphics Forum*, 18(4) :237–248, 1999. 1

[GC95] Steven J. Gortler and Michael F. Cohen. Hierarchical and variational geometric modeling with wavelets. In *Proceedings of the 1995 symposium on Interactive 3D graphics*, pages 35–42. ACM Press, 1995. 133

[GLDH97] M. H. Gross, L. Lippert, R. Dittrich, and S. Häring. Two methods for wavelet-based volume rendering. *Computers and Graphics*, 21(2) :237–252, 1997. 5

[GOMP98] Carlos Gonzalez-Ochoa, Scott McCammon, and Jörg Peters. Computing moments of objects enclosed by piecewise polynomial surfaces. *Transactions on Graphics*, 17(3) :143–157, 1998. 39, 69, 133

[GSCH93] Steven J. Gortler, Peter Schröder, Michael F. Cohen, and Pat Hanrahan. Wavelet radiosity. *Computer Graphics*, 27(Annual Conference Series) :221–230, 1993. 5

[GW73] Byron S. Gottfried and Joel Weisman. *Introduction to Optimization Theory*. Prentice Hall, Englewood Cliffs, New Jersey, 1973. 22, 120

[HBVMT99] Sunil Hadap, Endre Bangerter, Pascal Volino, and Nadia Magnenat-Thalmann. Animating wrinkles on clothes. In *VIS '99 : Proceedings of the conference on Visualization*, pages 175–182, 1999. 110

[HDD+94] Hugues Hoppe, Tony DeRose, Tom Duchamp, Mark Halstead, Hubert Jin, John McDonald, Jean Schweitzer, and Werner Stuetzle. Piecewise smooth surface reconstruction. *SIGGRAPH Computer Graphics*, pages 295–302, 1994. 133

[HHB93] H. Hagen, S. Hahmann, and G.P. Bonneau. Variational surface design and surface interrogation. *Computer Graphics Forum*, 12(3) :447–459, 1993. (Proceedings Eurographics'93). 120

[HKD93] Mark Halstead, Michael Kass, and Tony DeRose. Efficient, fair interpolation using Catmull-Clark surfaces. *Computer Graphics*, 27(Annual Conference Series) :35–44, 1993. 133

[Hop96] Hugues Hoppe. Progressive meshes. *SIGGRAPH Computer Graphics*, pages 99–108, 1996. 1

[JBU01] Mathews Jacob, Thierry Blu, and Michael Unser. An exact method for computing the area moments of wavelet and spline curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(06) :633–642, 2001. 133

[JJT05] Fu Jing, Ajay Joneja, and Kai Tang. Modeling wrinkles on smooth surfaces for footwear design. *Computer-Aided Design*, 37 :815–823, 2005. 110

[KCVS98] Leif Kobbelt, Swen Campagna, Jens Vorsatz, and Hans-Peter Seidel. Interactive multi-resolution modeling on arbitrary meshes. *Computer Graphics*, 32(Annual Conference Series) :105–114, 1998. 1

[KSS00] Andrei Khodakovsky, Peter Schröder, and Wim Sweldens. Progressive geometry compression. *SIGGRAPH Computer Graphics*, pages 271–278, 2000. 94

[Las87] John Lasseter. Principles of traditional animation applied to 3d computer animation. *SIGGRAPH Computer Graphics*, pages 35–44, 1987. 1, 110

[LC04] Caroline Larboulette and Marie-Paule Cani. Real-time dynamic wrinkles. In *Computer Graphics International*. IEEE Computer Society Press, 2004. Greece. 2, 110

[LDW97] M. Lounsbery, T. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Transaction on Graphics*, 16(1) :34–73, 1997. 1, 82, 84, 85, 92, 133

[LMQ01] T. Lyche, K. Mørken, and E. Quak. Theory and algorithms for non-uniform spline wavelets. In N. Dyn, D. Leviatan, D. Levin, and A. Pinkus, editors, *Multivariate Approximation and Applications*, pages 152–187. Cambridge University Press, 2001. 133

[Loo87] Loop. *Smooth subdivision surfaces based on triangles*. PhD thesis, University of Utah, Department of Mathematics, 1987. 82, 84, 88, 89

[LQS04] Denggao Li, Kaihuai Qin, and Hanqiu Sun. Unlifted loop subdivision wavelets. In *Pacific Graphics Conference on Computer Graphics and Applications*, pages 25–33, 2004. 82, 88, 92, 133

[Mal89] S. Mallat. A theory for multiresolution signal decomposition : the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11 :674–693, 1989. 5, 6

[Mer99] A. Mertins. *Signal Analysis : Wavelets, Filter Banks, Time-Frequency Transforms and Applications*. John Wiley & Sons, Chichester, 1999. 5

[Mit00] J. Mitchell. Geometric shortest paths and network optimization. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, volume 334, pages 633–702. Elsevier Science, 2000. 122

[MRMG] Caltech Multi-Res Modeling Group. http ://www.multires.caltech.edu/. 106, 107

[NR83] H. Nowacki and D. Reese. Design and fairing ship surfaces. *Surfaces in CAGD*, pages 121–134, 1983. 21, 119

[OBS04] Y. Ohtake, A. Belyaev, and H.-P. Seidel. Ridge-valley lines on meshes via implicit surface fitting. *ACM Transactions on Graphics*, 23(3) :609–612, 2004. 122

[PJF97] Q. Peng, X. Jin, and J. Feng. Arc-length-based axial deformation and length preserved animation. In *Computer Animation*, pages 86–92, 1997. 2

[PTBS01] Valérie Pham-Trong, Luc Biard, and Nicolas Szafran. Pseudo-geodesics on three-dimensional surfaces and pseudo-geodesic meshes. *Numer. Algorithms*, 26(4) :305–315, 2001. 122

[PVTF02] William H. Press, William T. Vetterling, Saul A. Teukolsky, and Brian P. Flannery. *Numerical Recipes in C++ : the art of scientific computing*. William T. Vetterling and Brian P. Flannery, 2002. 23, 121

[RP96] J.A. Roulier and B. Piper. Prescribing the length of rational bezier curves. *Computer-Aided Geometric Design*, 13(1) :23–43, 1996. 2

[RSB96] A. Rappoport, A. Sheffer, and M. Bercovier. Volume-preserving free-form solids. In *IEEE Transactions on Visualization and Computer Graphics*, volume 2(1), pages 19–27, 1996. 2

[SDS96] Eric J. Stollnitz, Tony DeRose, and David H. Salesin. *Wavelets for Computer Graphics : Theory and Applications*. Morgan Kaufmann Publishers, San Francisco, CA, USA, 1996. 5, 66, 67, 138

[Spa] SparseLib++. http ://math.nist.gov/sparselib++/. 100

[SS95] Peter Schröder and Wim Sweldens. Spherical wavelets : efficiently representing functions on the sphere. *SIGGRAPH Computer Graphics*, pages 161–172, 1995. 1, 82

[SS96] Wim Sweldens and Peter Schröder. Building your own wavelets at home. In *Wavelets in Computer Graphics, SIGGRAPH Course notes*, pages 15–87. ACM Press, 1996. 82, 88

[Sta98] Jos Stam. Evaluation of loop subdivision surfaces. In *SIGGRAPH Course notes*. ACM Press, 1998. 103

[Swe96] Wim Sweldens. The lifting scheme : A custom-design construction of biorthogonal wavelets. *Applied and Computational Harmonic Analysis*, 3(2) :186–200, 1996. 87

[Swe97] Wim Sweldens. The lifting scheme : A construction of second generation wavelets. *SIAM J. Math. Anal.*, 29(2) :511–546, 1997. 82, 86, 87

[TPBF87] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. *SIGGRAPH Computer Graphics*, pages 205–214, 1987. 2

[TQ94] D. Terzopoulos and H. Qin. Dynamic nurbs with geometric constraints for interactive sculpting. *Transactions on Graphics*, 13(2) :103–136, 1994. 2

[VP04] Sébastien Valette and Rémy Prost. Wavelet-based multiresolution analysis of irregular surface meshes. *IEEE Transactions on Visualization and Computer Graphics*, 10(2) :113–122, march/april 2004. 1

[VY92] Marie-Luce Viaud and Hussein Yahia. Facial animation with wrinkles. In *Eurographics Workshop on Animation and Simulation*, 1992. 110

[WKMMT99] Yin Wu, Prem Kalra, Laurent Moccozet, and Nadia Magnenat-Thalmann. Simulating wrinkles and skin aging. *The visual computer*, 15(4) :183–198, 1999. 110

[WKMT96] Yin Wu, Prem Kalra, and Nadia Magnenat-Thalmann. Simulation of static and dynamic wrinkles of skin. In *CA '96 : Proceedings of the Computer Animation*, page 90, Washington, DC, USA, 1996. IEEE Computer Society. 110

[WW92] William Welch and Andrew Witkin. Variational surface modeling. *SIGGRAPH Computer Graphics*, pages 157–166, 1992. 21, 119

[YHB05] Alex Yvart, Stefanie Hahmann, and Georges-Pierre Bonneau. Hierarchical triangular splines. *ACM Transactions on Graphics (TOG)*, 24(4) :1–18, 2005. 133

[ZSS97] Denis Zorin, Peter Schröder, and Wim Sweldens. Interactive multiresolution mesh editing. *SIGGRAPH Computer Graphics*, pages 259–268, 1997. 1