Université de Montréal

Comparaison de méthodes de détection automatique d'intersections sur surfaces paramétriques

par
Étienne Léger

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté à la Faculté des études supérieures
en vue de l'obtention du grade de Maître ès sciences (M.Sc.)
en informatique

Novembre, 2016

RÉSUMÉ

La question de déterminer si un modèle géométrique a des intersections non prévues est commune à plusieurs domaines : simulations numériques, CAO/DAO, animation, infographie, etc. C'est un problème dont la complexité varie avec la représentation choisie pour créer le modèle. Pour les surfaces paramétriques c'est un problème difficile à résoudre, mais pour lequel plusieurs solutions ont été proposées. Ces solutions diffèrent les unes des autres dans leurs angles d'approche, leur complexité et la justesse de leurs résultats. Dans ce mémoire, nous tenterons de comparer certaines de ces méthodes. Nous nous concentrerons sur les méthodes dites *failsafe*, c'est-à-dire qui permettent assurément de détecter la possibilité d'une intersection s'il y en a une. Ces méthodes sont celles utilisées pour toutes les applications critiques, donc pour lesquelles un modèle mal formé aurait des conséquences importantes.

Ce mémoire est à teneur principalement théorique. Nous comparerons les méthodes, dans un premier temps, sur leur puissance de résolution. Nous discuterons, dans un deuxième temps, de coût calculatoire. Nous avons finalement fait quelques implémentations pour appuyer nos observations théoriques, mais nous n'avons pas fait une analyse empirique approfondie des coûts calculatoires. Ceci reste à faire.

Nous verrons entre autre qu'il existe un ordre partiel entre certaines des méthodes, mais pas toutes. Par exemple, la méthode test-point est strictement plus puissante que la séparation des enveloppes convexes, mais elle est ni plus ni moins puissante que la méthode Volino-Thalmann.

Mots clés: détection, interférence, intersection, auto-intersection, surface, paramétrique, Bézier, subdivision.

ABSTRACT

The question of determining if a given geometric model has extraneous intersections is common to many domains: numerical simulation, CAD/CAM, animation, computer graphics, etc. The complexity of this problem varies with the representation chosen to generate the model. For parametric surfaces, it is a hard problem, but for which many solutions have been proposed. Those solutions differ from one another by their underlying ideas, their complexity and the exactitude of the result they give. In this thesis, we will try to compare some of these methods. We will concentrate on the class of methods we call failsafe, the methods that will surely detect the possibility of an intersection if there is one. Those are the methods used in all critical applications, the applications in which a malformed model would have important consequences.

The work of this thesis is mostly theoretical. First, we will compare the different techniques on their power of resolution. Then, we will discuss the execution cost of the different methods. We did some implementations while working on this thesis, but only as a way to support our theoretical observations. A complete empirical study of the execution times of the different methods would be left to do.

We will see that there is a partial order between some of the methods in their strength, but not all of them. For example, the test-point method is strictly stronger than the separation of the convex hulls method, but is neither stronger nor weaker than the Volino-Thalmann method.

Keywords: detection, interference, intersection, auto-intersection, parametric, surface, Bézier, subdivision.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES ANNEXES

LISTE DES SIGLES

AABB	Axis Aligned Bounding Box
BVH	Bounding Volume Hierarchy
BSP-tree	Binary Space Partition Tree
CAO/DAO	Conception Assistée par Ordinateur / Dessin Assisté par Ordinateur
	ou CAD/CAM, Computer Aided Design / Computer Aided Modelling, en anglais
NURBS	Non-Uniform Rational B-Spline
OBB	Oriented Bounding Box

NOTATION

b	un scalaire
$\boldsymbol{R}$	un vecteur
$\{\boldsymbol{Q}\}$	un ensemble de vecteurs
$\Theta(n)$	ordre de n (au sens de borne supérieure et inférieure stricte, la définition peut être trouvée dans [10, p.24], par exemple)
$\sum_{i+j+k=n}$	somme des éléments pour lesquels la somme des indices i, j et k vaut n
C^n	opérateur de de Casteljau appliqué n fois

In memoriam Gerald Farin.

REMERCIEMENTS

J'aimerais remercier, pour avoir influencé positivement d'une manière ou d'une autre le contenu de ce mémoire : Neil Frederick Stewart, Lars-Erik Andersson, Lise Audet, Jean-Pierre Léger, Gabrielle Doré, Hélène Gaudreau, Michael Meaney, Josie Diorio, Soline Asselin, Philippe Richard, Léa Audet, Renaud Besner, Nassim Rousset, Jérémy Doucet-Koussaya, Jonathan Bonneau, Morgan Fortier, Pierre Poulin, Derek Nowrouzezahrai, Tiberiu Popa, Paul Charbonneau, Daniel Nadeau, Guillaume-Olivier Choquette, Sonia Abdoli, David Quiroz Marin, Olivier Mercier, Cynthia Beauchemin, Gilles-Philippe Paillé, David Milaenen, Chaitanya Chakravarty, Joël Polard-Perron, Bruno Roy, Melino Conte, Luis Gamboa, Adrien Dubouchet, Alexandre Jubert, Jean-Philippe Guertin-Renaud, Yangyang Zhao, Cihan Özer, Arnaud Schoentgen, Antoine Webanck, Aude Giraud, Olivier Mastropietro, Irina Pokhvisneva, Paula González Treviño, Valérie Aubut, Cassandre Carpentier Laberge, Camille Gagnon-Trudeau, Erika Piercy, Caroline Boivin, Jessica Bernard, Hubert Reeves, Max Planck, Albert Jacquard, René Descartes, Jean-Paul Sartre, Gottfried Wilhelm von Leibniz, ainsi que tous les hommes et toutes les femmes de science qui ont pavé la voie avant moi.

CHAPITRE 1

INTRODUCTION

La détection d'intersections non désirées dans des modèles géométriques est un sujet de recherche actif depuis plus de 60 ans. Il n'existe pourtant toujours pas de solution simple et unique, quelle que soit la situation d'intérêt. C'est aussi un problème important puisqu'il est nécessaire dans une variété de domaines, citons, entre autres, les jeux vidéo, l'animation 3D et les différentes branches du CAO/DAO. Évidemment, sa complexité croît avec le degré des surfaces étudiées. Pour des surfaces de degré 1, les maillages polygonaux, le problème peut être résolu de façon exacte, sauf erreurs d'arrondis. Non seulement il est possible de trouver la solution exacte, mais il est même possible de le faire assez efficacement en utilisant certaines structures d'accélération, les *BVH* par exemple (voir section 2.4 pour plus de détails sur ces structures). Par contre, pour les surfaces plus générales, le problème ne peut pas être résolu exactement sans devoir trouver les racines de polynômes de degrés élevés, ce qui est difficile et sujet aux instabilités numériques.

Les maillages polygonaux sont préférés dans une variété de domaines de par leur simplicité. Or, pour plusieurs autres applications, en CAO/DAO surtout, pour le *CNC milling* notamment, une plus grande précision et flexibilité de modélisation sont nécessaires ; les surfaces paramétriques sont alors le meilleur choix. Dans d'autres contextes, la carosserie ou l'avionique, par exemple, il est souvent aussi nécessaire d'avoir des surfaces appartenant à un ordre de continuité plus élevé. Dans ces cas aussi les surfaces paramétriques seront préférables. Les surfaces de Bézier cubiques, par exemple, ont un niveau de continuité paramétrique C^2, et il est possible d'assurer la même continuité entre des surfaces adjacentes en ajoutant certaines contraintes sur les points de contrôle. Les surfaces paramétriques ont donc une utilité pratique certaine et l'intérêt de trouver une solution au problème de détection d'intersection est grand. Or, puisque l'obtention d'une solution exacte est trop complexe et trop

coûteuse[1], il faut trouver des méthodes pour estimer. Ces estimations doivent être les plus précises, au sens de forte puissance de résolution possible, tout en étant les moins chères possible. En plus d'optimiser le coût et la puissance de détection d'intersection, nous exigeons aussi, de manière générale, que ces méthodes soient *failsafe*, c'est-à-dire que l'algorithme doit toujours signaler la possibilité d'une intersection s'il y en a une. Cette propriété est exigée puisqu'il est normalement de loin préférable d'être conservateur en déclarant la possibilité d'une intersection qui n'existe pas que de ne pas repérer une intersection véritable, les conséquences de l'une et de l'autre étant bien différentes. Pour la première, il s'agit d'un coût de calcul supplémentaire, alors que pour la deuxième, c'est généralement la production d'un modèle mal formé, ce qui peut avoir des conséquences beaucoup plus importantes.

1.1 Objectif de ce mémoire

L'objectif de ce mémoire n'est pas d'implémenter l'algorithme automatique de détection d'intersection global le plus efficace, mais c'est tout de même dans ce contexte que les méthodes seront envisagées. Nous nous intéresserons à l'utilisation des différentes méthodes dans le contexte d'une automatisation complète. Nous devrons donc, pour les tests qui nécessitent certaines conditions préalables, pouvoir les vérifier par des processus automatiques avant de faire le test.

Nous allons comparer entre elles les méthodes de détection d'intersection sur les surfaces paramétriques, au niveau le plus détaillé. Dans la pratique, ces méthodes sont les dernières utilisées dans le processus de comparaison global, qui fait généralement appel à une structure hiérarchique et à la subdivision, puisque les surfaces sur lesquelles ces algorithmes travaillent sont généralement composées. Ces méthodes sont celles que l'on appelle lorsqu'une intersection ne peut pas être exclue entre deux sous-surfaces. Ceci se produit donc lorsque les deux sous-surfaces en question sont proches et/ou que la normale change beaucoup, ou alors lorsque deux sous-surfaces partagent une courbe frontière. Dans ce cas, les techniques pour exclure l'intersection sans passer

[1] Comme montré par Sederberg [29, p.61], la courbe d'intersection entre deux surfaces Bézier cubiques est un polynôme de degré 324, puisqu'il correspond à $(2d^2)^2$, où d est le degré de la surface.

explicitement au test ne fonctionnent pas. Dans les tests pratiques que nous ferons, nous choisirons des surfaces qui correspondent à ces critères. Or, ce mémoire vise d'abord à comparer de manière théorique. Ces tests serviront alors principalement de support et de vérification *ad hoc* pour les démonstrations que nous ferons plutôt que de servir de résultats pratiques véritablement réalistes. Nous reviendrons sur ces tests au chapitre 4.

Dans les algorithmes globaux, l'appel à des méthodes de subdivision et le choix judicieux de l'utilisation de méthodes plus ou moins coûteuses à différents niveaux de la subdivision valent pour une grande partie dans la différence d'efficacité. Ce mémoire vise à faire la lumière sur les différences entre les divers tests afin de se donner les outils nécessaires pour être en mesure de faire ce choix lors de la conception d'un tel algorithme global. Les surfaces paramétriques étant très utilisées en ingénierie et en sciences appliquées, l'intérêt de mieux comprendre les outils disponibles est donc grand. En effet, les surfaces paramétriques sont souvent utilisées puisqu'elles offrent plusieurs avantages sur les surfaces implicites[2]. Nous pourrions mentionner par exemple :

- Une plus grande facilité de représentation pour les surfaces bornées.
- Une plus grande facilité à calculer un point donné de la surface.
- Un ordonnancement plus naturel lors de la génération de points suivant un maillage sur la surface.
- Une représentation plus intuitive géométriquement, permettant la création d'outils de modélisation plus faciles d'utilisation.

Or, elles ont aussi des désavantages par rapport à leurs vis-à-vis implicites :

[2] Les surfaces implicites et paramétriques sont deux représentations différentes pour modéliser des courbes ou des surfaces. Elles permettent généralement de modéliser les mêmes objets. Or, de manière générale, la représentation est plus facile à construire ou à manipuler dans une forme que dans l'autre. Prenons une sphère de rayon 1 centrée à l'origine, par exemple. Celle-ci peut être représentée implicitement par la fonction $x^2+y^2+z^2-1=0$, mais elle peut aussi être représentée paramétriquement par $S(x(u,v), y(u,v), z(u,v))$
où
$x(u,v) = \sin(u)\cos(v)$
$y(u,v) = \sin(u)\sin(v)$
$z(u,v) = \cos(u)$
avec $0 \leq u \leq \pi,\ 0 \leq v \leq 2\pi$.

- Une plus grande difficulté à représenter des surfaces non bornées.
- Une plus grande difficulté à déterminer si un point de l'espace appartient à la surface.
- L'existence de points critiques dans la paramétrisation. L'exemple donné dans la note de bas de page 2 permet d'illustrer ce problème. Dans la forme paramétrique, la normale est mal définie aux pôles, bien que ces points soient géométriquement identiques à tout autre point de la sphère.

Pour une présentation détaillée des surfaces paramétriques, le lecteur intéressé peut consulter [26], et pour plus de détails sur les surfaces implicites, les autres représentations et les différences entre celles-ci [20].

Considérant leurs avantages plus importants et leur plus grande popularité, nous ferons le choix de nous concentrer sur les surfaces paramétriques. De plus, notre étude vise principalement à distinguer les méthodes selon leur puissance de discrimination. Nous discuterons aussi de coût, au chapitre 4, mais uniquement à titre indicatif. Une analyse approfondie des coûts calculatoires réels reste à faire afin d'avoir un portrait global de la situation quant à l'applicabilité dans les systèmes existants. Finalement, notons que nous ne nous intéresserons qu'aux méthodes de détection d'intersection de surfaces paramétriques entre elles et d'auto-intersection. Nous n'étudierons pas le problème connexe de détecter des intersections entre une surface paramétrique et un autre objet. Ce problème étant plus large, il laisse place à une littérature abondante. Nous pourrions citer, par exemple, le problème de trouver l'intersection entre une surface paramétrique et un rayon, dans le cadre de *ray-tracing* [14] [25], etc.

1.2 Représentation, notation et conversion

Dans ce mémoire, nous étudierons principalement les surfaces de Bézier triangulaires. Ce choix a dû être fait puisqu'il fallait une représentation commune pour mettre les méthodes en relation. Or, cette représentation n'est pas limitante en ce sens qu'il est possible par conversion de représentation d'appliquer les différentes méthodes à

d'autres types de surfaces. Par exemple, il est possible de passer d'une surface Bézier produit tensoriel à une surface Bézier triangulaire, comme montré dans [16]. Il est aussi possible de passer d'une surface B-spline à une surface Bézier en utilisant l'algorithme de Böhm, comme expliqué dans [7]. De même, nous pouvons aussi passer d'une surface de subdivision de Loop, à condition d'éviter les sommets extraordinaires, à une surface de Bézier en utilisant les matrices de transformation données dans [28, p.24] ou la paramétrisation donnée dans [32]. Le choix des surfaces de Bézier triangulaires est aussi motivé par plusieurs de leurs propriétés qui nous seront utiles pour la construction de preuves. Le lecteur intéressé peut lire [12] pour une présentation détaillée de ces surfaces. Tous les changements de représentation énoncés plus haut sont possibles puisque les surfaces de Bézier sont en réalité un cas particulier des B-splines, qui sont elles-mêmes un cas particulier des NURBS (Non-Uniform Rational B-Splines). Les B-splines sont aussi un cas particulier des Box-splines dont font partie les surfaces de subdivision de Loop, comme expliqué dans [4].

Avant de se pencher sur la présentation des différentes méthodes, il est important de définir quelques concepts et de choisir une notation commune. Cela est nécessaire puisque chaque méthode, dans sa formulation originale, est présentée avec sa propre terminologie. Nous utiliserons une notation proche de celle présentée dans [2]. Ainsi, nous utiliserons $\boldsymbol{Q} = \{\boldsymbol{R}_0, \boldsymbol{R}_1, \boldsymbol{R}_2, ..., \boldsymbol{R}_n\}$ pour représenter la séquence des points de contrôle d'une courbe Bézier et $\boldsymbol{Q} = \{\boldsymbol{R}_{ijk}\}$ avec $i, j, k \geq 0$ et $i + j + k = n$ pour l'ensemble des points de contrôle définissant une surface.

La surface de Bézier triangulaire associée à ces points de contrôle est définie comme :

$$(r, s, t) \mapsto \sum_{i+j+k=n} \frac{n!}{i!j!k!} r^i s^j t^k \boldsymbol{R}_{ijk} \tag{1.1}$$

où r, s et t sont des coordonnées barycentriques, donc $r, s, t \geq 0$ et $r + s + t = 1$. De manière plus algorithmique, la surface générée par les points de contrôle peut aussi être définie en utilisant l'opérateur de de Casteljau :

$$(r, s, t) \mapsto C^n(r, s, t)\boldsymbol{Q} \tag{1.2}$$

avec l'opérateur de de Casteljau défini comme :

$$C(r,s,t) = rE_{100} + sE_{010} + tE_{001}$$

où E_{100}, E_{010} et E_{001} sont les opérateurs de décalage dans les directions i, j, et k respectivement :

$$E_{100}\{\boldsymbol{R}_{ijk}\}_{i,j,k\geq 0;\, i+j+k=n} = \{\boldsymbol{R}_{i+1,j,k}\}_{i,j,k\geq 0;\, i+j+k=n-1}$$

avec E_{010} et E_{001} définis de manière analogue. Cette façon plus algorithmique de définir la surface sera très utile, comme nous le verrons plus loin. L'opérateur de de Casteljau est central pour manipuler les courbes et les surfaces de Bézier puisqu'il permet à la fois d'évaluer et de subdiviser la surface. De plus, la subdivision est quasi nécessaire pour toute méthode de haut niveau opérant sur des surfaces de Bézier, comme nous le verrons à la sous-section 2.4.2. Elle permet de fractionner le problème et d'accélérer la recherche en ne se concentrant que sur les parties de la surface où il y a risque d'intersection non désirée. Cet opérateur nous servira aussi à construire les ensembles de différences de points de contrôle utilisés dans les méthodes *test-point*.

1.3 Plan du mémoire

La suite du mémoire sera divisée en quatre chapitres :

- Le chapitre 2 présente un résumé des principales méthodes existantes pour détecter la présence d'intersections sur des surfaces paramétriques.
- Le chapitre 3 présente des résultats nouveaux et jamais publiés à ce jour sur les différences théoriques entre les méthodes quant à leur puissance de discrimination.
- Le chapitre 4 présente une discussion sur les différences de coût calculatoire des méthodes étudiées.
- Le chapitre 5 présente, en guise de conclusion, une liste de problèmes ouverts sur le sujet, de questions connexes et de directions de recherche future possibles.

CHAPITRE 2

PRÉSENTATION DES MÉTHODES

Plusieurs méthodes ont été proposées pour détecter les intersections entre des surfaces paramétriques. Ces méthodes diffèrent dans leur implémentation, mais aussi et surtout dans leur puissance de discrimination[1], leur coût d'utilisation et la robustesse de leur résultat. Voici donc un survol des différentes techniques et une description de leurs caractéristiques propres.

2.1 Volumes englobants

Cette méthode est la plus simple de toutes les méthodes utilisées. Justement, son intérêt réside principalement dans sa simplicité. C'est pour cette raison qu'elle est quasi toujours utilisée dans les structures hiérarchiques de haut niveau. D'ailleurs, plusieurs structures de haut niveau ne font appel qu'à ce type de méthodes, citons [17] par exemple. Nous reviendrons sur ces structures d'accélération à la section 2.4.

Cette méthode est facilement implémentable et demande très peu de temps de calcul. Sa puissance de résolution est toutefois très limitée. Elle ne peut pas être utilisée pour détecter les cas d'auto-intersection ni les intersections entre les surfaces qui partagent une arête en commun, mais seulement les intersections entre des objets distincts. Elle consiste à fabriquer des volumes englobants autour des géométries considérées, puis à vérifier que ces volumes englobants sont séparables. Le volume englobant peut être n'importe quelle géométrie convexe, *e.g.*, un parallélépipède rectangulaire, une sphère [8], un ellipsoïde [22], etc.

[1] Il est à noter que puisque les méthodes présentées sont des estimateurs, des conditions suffisantes seulement, la puissance prend ici le sens de capacité à déterminer que deux surfaces sont séparables. Ainsi, une méthode dite plus puissante qu'une autre sera capable de déterminer qu'il n'y a pas d'intersections non désirées dans plus de cas. S'il y a effectivement une intersection, puisque les méthodes étudiées sont toutes *failsafe*, les deux méthodes donneront le même résultat, soit qu'il y a possibilité d'intersection. Dans le texte, les termes « puissance » et « précision » seront tous deux utilisés dans ce sens. Autrement dit, pour utiliser les termes de la classification binaire et de la recherche d'information, toutes les méthodes comparées auront un rappel (ou *true positive rate*) de 1, seule la précision varie d'une méthode à l'autre (voir [21], par exemple, pour les définitions formelles du rappel et de la précision).

Évidemment, plus le volume englobant choisi est complexe, plus il est possible d'encadrer étroitement l'objet. Or, plus le volume englobant est complexe, plus la construction de ce volume est difficile et plus le test de l'intersection entre deux de ces volumes est difficile. Il existe donc diverses déclinaisons de cette technique avec des volumes plus ou moins complexes et donnant des résultats plus ou moins précis. Comme l'avantage de ces techniques réside dans leur simplicité sur le plan des calculs et non dans leur puissance de résolution élevée, la géométrie englobante la plus simple est généralement utilisée, *i.e.*, le parallélépipède rectangulaire. Dans ce cas, les intersections ne demandent que la résolution d'équations linéaires, plutôt que quadratiques comme pour la sphère et l'ellipsoïde par exemple. Ce parallélépipède rectangulaire peut être orienté dans l'espace de différentes façons. Lorsque les côtés de cette boîte sont choisis pour coïncider avec les axes de l'espace, nous avons la technique *Axis Aligned Bounding Boxes* (AABB), alors que si nous choisissons une orientation arbitraire de la boîte, nous avons plutôt la technique *Oriented Bounding Boxes* (OBB). Encore une fois, l'OBB permet en général d'avoir un volume englobant plus petit, donc d'obtenir plus de précision, mais en contrepartie, elle demande de résoudre des équations linéaires pour trouver les intersections, alors que pour l'AABB, il ne suffit que de tester des inégalités avec les coins de la boîte. La méthode AABB est très utilisée sur les surfaces polygonales et dans le cadre de structures d'accélération.

Or, ce qui nous intéressera dans le cadre de ce mémoire sera plutôt les surfaces paramétriques, les surfaces de Bézier en particulier. Dans ce cas, la géométrie englobante convexe la plus précise et dont la construction est aussi très simple est l'enveloppe convexe des points de contrôle, utilisant la méthode présentée dans [6] par exemple. Le test d'intersection avec celui-ci sera donc toujours plus puissant que celui avec n'importe lequel des volumes vus jusqu'à maintenant, puisque les autres volumes englobants sont également convexes, voir la figure 2.1. Il est possible d'utiliser l'enveloppe convexe puisqu'une surface de Bézier est toujours circonscrite dans l'enveloppe de ses points de contrôle, comme montré dans [13]. C'est d'ailleurs cette propriété qui permettait également de construire les autres volumes englobants vus jusqu'à maintenant. Cette technique peut, par conséquent, être utilisée pour toutes les surfaces ayant cette propriété,

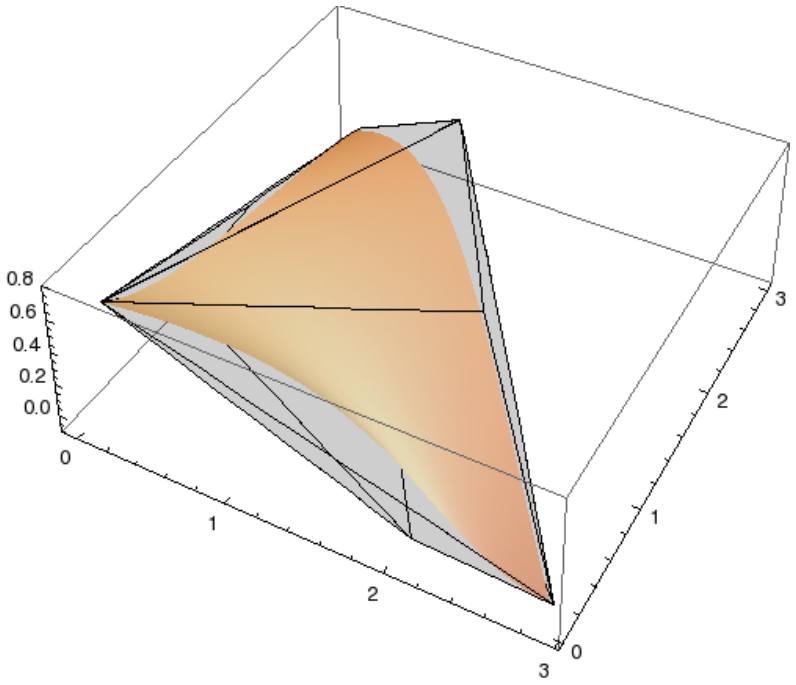

Figure 2.1 – Comparaison de volumes englobants. En orange : la surface de Bézier ; en gris : l'enveloppe convexe des points de contrôle ; en blanc : la boîte englobante alignée avec les axes.

les B-splines ou les NURBS, par exemple. L'enveloppe convexe est simple à construire, par contre, encore une fois, cette méthode ne peut que traiter les cas d'intersection entre objets distincts et non celui de l'auto-intersection. Ce volume est plus précis que ceux vus précédemment, mais son coût calculatoire est conséquemment plus élevé. En effet, pour la méthode AABB, la construction du volume prenait $\Theta(n)$ temps et le test d'intersection des boîtes $\Theta(1)$, où n est le nombre de points de contrôle de la surface. Pour l'enveloppe convexe, même en utilisant les algorithmes les plus efficaces, *Quickhull* par exemple [6], la construction demande $\Theta(n \lg n)$ en moyenne. De même, le calcul pour vérifier la non-intersection des enveloppes convexes peut se faire en $\Theta(n)$ en moyenne, où n est le nombre total de points formant les deux enveloppes, comme montré dans [9]. Nous voyons donc que cette méthode coûte passablement plus cher que AABB. Or, dans le contexte des surfaces paramétriques, la précision a généralement une importance plus grande que pour les surfaces polygonales. Cette méthode a de ce fait une réelle importance pratique. Elle nous servira de *baseline* pour les comparaisons sur la puissance

des méthodes au chapitre 3.

2.2 Méthodes *test-point*

La classe de méthodes que nous appellerons *test-point* sont les méthodes qui construisent un ensemble de points comme combinaison des points de contrôle originaux pour tester l'intersection, contrairement aux méthodes précédentes qui opéraient directement sur les points de contrôle. Cette classe de méthodes a plus de flexibilité que la méthode précédente : elle permet de tester l'intersection d'un objet avec lui-même, entre des objets distincts et entre des objets adjacents. Les tests d'intersection entre objets adjacents, soit deux surfaces avec une courbe frontière commune ou deux surfaces ayant un point de contrôle en commun, sont d'une importance pratique certaine. Ces cas sont très fréquemment rencontrés puisqu'ils se présentent lors de la comparaison des différents morceaux d'une surface après subdivision et que ce sont les seuls cas possibles pour un complexe simplicial bien formé, voir [19, p.202] pour une définition formelle des complexes simpliciaux curvilinéaires. De plus, la sous-méthode pour les objets distincts a plus de puissance de résolution que la méthode précédente, comme nous le verrons à la section 3.1.

2.2.1 Auto-intersection

L'idée derrière les méthodes test-point est de construire une fonction qui sera égale à zéro si et seulement si la surface a une auto-intersection, c'est-à-dire si la surface a la même valeur pour deux points distincts dans l'espace paramétrique ou si la dérivée directionnelle de l'application $(r, s, t) \mapsto \boldsymbol{R}(r, s, t)$ est égale au vecteur nul, en un point. Ce dernier cas peut être considéré comme le cas limite d'une auto-intersection véritable. La fonction que nous dériverons ne sera pas utilisée explicitement dans le test d'auto-intersection, mais elle sera utile à la compréhension de l'origine des ensembles de test-point que nous utiliserons pour le test. La dérivation de cette fonction est une traduction plus ou moins telle quelle de l'article original [3, Partie II, §2.1]. Pour plus de détails, le lecteur peut donc s'y référer. Il trouverait peut-être aussi utile la lecture de

[3, Partie I, §2.1] qui montre une dérivation semblable pour les courbes de Bézier. Cette dernière permet de voir plus facilement l'idée derrière la preuve dans un cas plus simple.

Commençons par définir le triangle suivant :

$$\mathcal{U} = \{\boldsymbol{P} = r\boldsymbol{\rho} + s\boldsymbol{\sigma} + t\boldsymbol{\tau} : r + s + t = 1,\ r, s, t \geq 0\}, \tag{2.1}$$

où r, s, t, comme nous l'avons précédemment défini, sont des coordonnées barycentriques et $\boldsymbol{\rho}$, $\boldsymbol{\sigma}$ et $\boldsymbol{\tau}$ sont des vecteurs quelconques, *i.e.*, affinement indépendants. Utilisant cette notation, nous pouvons donc écrire $\boldsymbol{R}(r, s, t)$ comme $\boldsymbol{R}(\boldsymbol{P})$. Ainsi, plus formellement, nous cherchons à construire une fonction qui sera zéro si et seulement si il existe $\boldsymbol{P}_0, \boldsymbol{P}_1$ tels que $\boldsymbol{P}_0 \neq \boldsymbol{P}_1$ et $\boldsymbol{R}(\boldsymbol{P}_0) = \boldsymbol{R}(\boldsymbol{P}_1)$ ou alors s'il existe un $\boldsymbol{P}_0$ où la dérivée directionnelle de l'application est égale à $\mathbf{0}$ (voir [12, p.86] pour une explication sur la signification de la dérivée directionnelle dans le contexte de fonctions à variables barycentriques).

Supposons que nous ayons bel et bien deux points $\boldsymbol{P}_0$ et $\boldsymbol{P}_1$ pour lesquels $\boldsymbol{P}_0 \neq \boldsymbol{P}_1$ et $\boldsymbol{R}(\boldsymbol{P}_0) = \boldsymbol{R}(\boldsymbol{P}_1)$ et regardons la droite passant par $\boldsymbol{P}_0$ et $\boldsymbol{P}_1$. Il y a une autre droite L' parallèle à cette dernière qui passe soit par :

a) le point $\boldsymbol{\rho}$ et intersectant le côté opposé $\boldsymbol{\sigma\tau}$

b) le point $\boldsymbol{\sigma}$ et intersectant le côté opposé $\boldsymbol{\tau\rho}$

c) le point $\boldsymbol{\tau}$ et intersectant le côté opposé $\boldsymbol{\rho\sigma}$

(voir la figure 2.2). Les trois mêmes cas se présentent pour la dérivée directionnelle nulle sur une droite reliant un point $\boldsymbol{P}_0$ et les coins du triangle.

Regardons chacun des trois cas séparément. Commençons par le cas a). Sans perte de généralité, nous pouvons assumer que $r_0 > r_1$. D'ailleurs, puisque $r_0 - r_1 = (s_1 - s_0) + (t_1 - t_0),\ s_1 - s_0 \geq 0$ et $t_1 - t_0 \geq 0$, nous pouvons écrire

$$\boldsymbol{P}_1 - \boldsymbol{P}_0 = (r_1 - r_0)\boldsymbol{\rho} + (s_1 - s_0)\boldsymbol{\sigma} + (t_1 - t_0)\boldsymbol{\tau} = (s_1 - s_0)(\boldsymbol{\sigma} - \boldsymbol{\rho}) + (t_1 - t_0)(\boldsymbol{\tau} - \boldsymbol{\rho}). \tag{2.2}$$

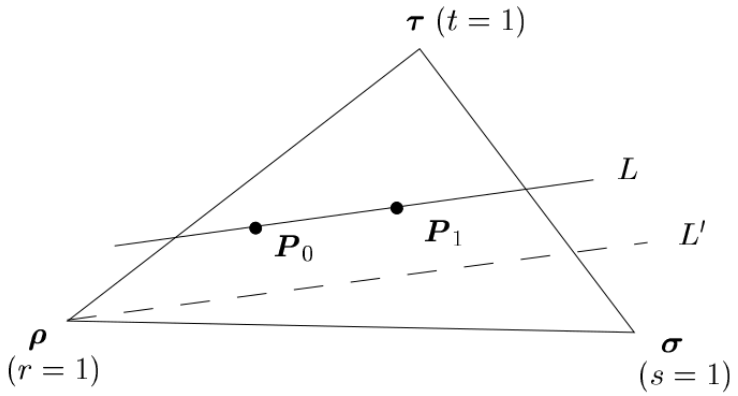

Figure 2.2 – Dessin tiré de [3] montrant le cas a) pour L'.

Définissons aussi

$$\alpha = \frac{t_1 - t_0}{r_0 - r_1},\ 1 - \alpha = \frac{s_1 - s_0}{r_0 - r_1},$$

qui peuvent être vus comme les coefficients permettant d'écrire la direction comme la moyenne entre les directions $\boldsymbol{\sigma} - \boldsymbol{\rho}$ et $\boldsymbol{\tau} - \boldsymbol{\rho}$. Nous pouvons alors réécrire

$$\boldsymbol{P}_1 - \boldsymbol{P}_0 = (r_0 - r_1)\Big((1-\alpha)(\boldsymbol{\sigma} - \boldsymbol{\rho}) + \alpha(\boldsymbol{\tau} - \boldsymbol{\rho})\Big), 0 \leq \alpha \leq 1, r_0 > r_1. \tag{2.3}$$

De la même façon, nous pouvons aussi définir $\alpha_0 = \frac{t_0}{1-r_0}$ lorsque $r_0 \neq 1$ et α_0 quelconque lorsque $r_0 = 1$. Ceci nous permet alors de réécrire

$$\boldsymbol{P}_0 - \boldsymbol{\rho} = (1 - r_0)\Big((1-\alpha_0)(\boldsymbol{\sigma} - \boldsymbol{\rho}) + \alpha_0(\boldsymbol{\tau} - \boldsymbol{\rho})\Big), 0 \leq \alpha_0 \leq 1. \tag{2.4}$$

De manière analogue à l'équation (2.3), nous pouvons aussi écrire pour l'opérateur de de Casteljau

$$C(\boldsymbol{P}_1) - C(\boldsymbol{P}_0) = (r_0 - r_1)\Big((1-\alpha)(E_{010} - E_{100}) + \alpha(E_{001} - E_{100})\Big), \tag{2.5}$$

ou plus simplement

$$C(\boldsymbol{P}_1) - C(\boldsymbol{P}_0) = (r_0 - r_1)C(-1, 1-\alpha, \alpha). \tag{2.6}$$

Ainsi, il est possible de réécrire le vecteur de différence entre deux points $\boldsymbol{R}(\boldsymbol{P}_1) - \boldsymbol{R}(\boldsymbol{P}_0)$ comme

$$\begin{aligned}\boldsymbol{R}(\boldsymbol{P}_1) - \boldsymbol{R}(\boldsymbol{P}_0) &= (C^n(\boldsymbol{P}_1) - C^n(\boldsymbol{P}_0))\boldsymbol{Q} \\ &= \left(\sum_{k=0}^{n-1} C^{n-1-k}(\boldsymbol{P}_1)C^k(\boldsymbol{P}_0)\right) \cdot (C(\boldsymbol{P}_1) - C(\boldsymbol{P}_0))\boldsymbol{Q}.\end{aligned} \tag{2.7}$$

Nous pouvons donc définir la fonction

$$\begin{aligned}\boldsymbol{S}_a(\boldsymbol{P}_0, \boldsymbol{P}_1) &\doteq \frac{1}{n}\frac{1}{(r_0 - r_1)}\Big(\boldsymbol{R}(\boldsymbol{P}_1) - \boldsymbol{R}(\boldsymbol{P}_0)\Big) \\ &= \frac{1}{n}\left(\sum_{k=0}^{n-1} C^{n-1-k}(\boldsymbol{P}_1)C^k(\boldsymbol{P}_0)\right) \cdot \left((1-\alpha)C(-1,1,0) + \alpha C(-1,0,1)\right)\boldsymbol{Q}.\end{aligned} \tag{2.8}$$

Cette fonction est définie sur $\{(\boldsymbol{P}_0, \boldsymbol{P}_1) \in \mathcal{U} \times \mathcal{U} : \boldsymbol{P}_0 \neq \boldsymbol{P}_1$ où $\boldsymbol{P}_0$ et $\boldsymbol{P}_1$ appartiennent au cas a) $\}$. En remarquant que le paramètre α dépend seulement de la direction de la différence $\boldsymbol{P}_1 - \boldsymbol{P}_0$, nous pouvons alors voir α comme un paramètre fixe et l'expression de $\boldsymbol{S}_a$ comme un polynôme de degré $n-1$ dans les deux variables 2-D $\boldsymbol{P}_1$ et $\boldsymbol{P}_0$. En gardant α fixe, et en faisant tendre $\boldsymbol{P}_1$ vers $\boldsymbol{P}_0$, nous obtenons :

$$\lim_{\boldsymbol{P}_1 \to \boldsymbol{P}_0} \boldsymbol{S}_a(\boldsymbol{P}_0, \boldsymbol{P}_1) = C^{n-1}(\boldsymbol{P}_0)\Big((1-\alpha)C(-1,1,0) + \alpha C(-1,0,1)\Big)\boldsymbol{Q}. \tag{2.9}$$

En remplaçant $\boldsymbol{P}_0$ et $\boldsymbol{P}_1$ par $(\alpha_0, r_0, \alpha, r_1)$, nous pouvons définir une nouvelle fonction $\tilde{\boldsymbol{S}}_a(\alpha_0, r_0, \alpha, r_1)$ égale à (2.8) lorsque $\boldsymbol{P}_0 \neq \boldsymbol{P}_1$ et (2.9) lorsque $\boldsymbol{P}_0 = \boldsymbol{P}_1$.

Nous voyons maintenant que $\boldsymbol{R}(\boldsymbol{P}_0) = \boldsymbol{R}(\boldsymbol{P}_1)$ si et seulement si $\boldsymbol{S}_a(\boldsymbol{P}_0, \boldsymbol{P}_1) = \tilde{\boldsymbol{S}}_a(\alpha_0, r_0, \alpha, r_1) = \boldsymbol{0}$. De plus, l'application $\boldsymbol{P} \mapsto \boldsymbol{R}(\boldsymbol{P})$ est singulière si et seulement si $\tilde{\boldsymbol{S}}_a(\alpha_0, r_0, \alpha, r_0) = \boldsymbol{0}$.

Il est possible de construire les fonctions analogues $\boldsymbol{S}_b$, $\tilde{\boldsymbol{S}}_b$, $\boldsymbol{S}_c$ et $\tilde{\boldsymbol{S}}_c$ pour les deux autres cas par permutation cyclique des variables r, s et t et des variables dans l'opérateur de de Casteljau. Une condition nécessaire et suffisante pour prouver que la surface de Bézier ne s'auto-intersecte pas est donc de vérifier que les fonctions $\tilde{\boldsymbol{S}}_a$, $\tilde{\boldsymbol{S}}_b$ et $\tilde{\boldsymbol{S}}_c$ ne s'annulent pas.

Cette condition étant difficile à tester, un test plus simple est proposé au [2, Criterion

3.1, p.517]. Il s'agit de remarquer que la surface $\tilde{\boldsymbol{S}}_a$ est une combinaison convexe des points de l'ensemble $\boldsymbol{q}_a$ et de même pour $\tilde{\boldsymbol{S}}_b$ et $\tilde{\boldsymbol{S}}_c$ avec $\boldsymbol{q}_b$ et $\boldsymbol{q}_c$:

$$\begin{aligned}
\boldsymbol{q}_a &= (C(-1,1,0)\boldsymbol{Q}) \cup (C(-1,0,1)\boldsymbol{Q}),\\
\boldsymbol{q}_b &= (C(0,-1,1)\boldsymbol{Q}) \cup (C(1,-1,0)\boldsymbol{Q}),\\
\boldsymbol{q}_c &= (C(1,0,-1)\boldsymbol{Q}) \cup (C(0,1,-1)\boldsymbol{Q}).
\end{aligned} \tag{2.10}$$

Le test est donc de vérifier que l'origine n'est incluse dans aucune des enveloppes convexes de ces trois ensembles.

Finalement remarquons que ces ensembles de test-point peuvent être réécrits comme :

$$\boldsymbol{q}_a = \boldsymbol{F}_{12} \cup \boldsymbol{F}_{13},\ \boldsymbol{q}_b = \boldsymbol{F}_{23} \cup \boldsymbol{F}_{21},\ \boldsymbol{q}_c = \boldsymbol{F}_{31} \cup \boldsymbol{F}_{32}, \tag{2.11}$$

où

$$\begin{aligned}
\boldsymbol{F}_{12} &= C(-1,1,0)\boldsymbol{Q}, \quad -\boldsymbol{F}_{12} = \boldsymbol{F}_{21} = C(1,-1,0)\boldsymbol{Q},\\
\boldsymbol{F}_{23} &= C(0,-1,1)\boldsymbol{Q}, \quad -\boldsymbol{F}_{23} = \boldsymbol{F}_{32} = C(0,1,-1)\boldsymbol{Q},\\
\boldsymbol{F}_{31} &= C(1,0,-1)\boldsymbol{Q}, \quad -\boldsymbol{F}_{31} = \boldsymbol{F}_{13} = C(-1,0,1)\boldsymbol{Q}.
\end{aligned} \tag{2.12}$$

Nous pouvons donc voir les ensembles $\boldsymbol{q}_\nu$ comme des différences des points de contrôle par rapport à chacune des directions du triangle paramétrique (voir la figure 2.3 qui illustre le cas $\boldsymbol{q}_a$).

Le test auquel nous référerons à partir de maintenant lorsque nous parlerons de méthode test-point pour l'auto-intersection est le dernier présenté, soit séparer les enveloppes convexes $\boldsymbol{q}_a,\ \boldsymbol{q}_b$ et $\boldsymbol{q}_c$ de l'origine.

2.2.2 Objets distincts

Pour tester l'intersection entre deux objets distincts, une méthode analogue à celle de l'auto-intersection peut être utilisée. Comme pour l'auto-intersection, les ensembles de test-point sont les points de contrôle d'une fonction construite en prenant les différences des points de contrôle originaux. La méthode de construction est tout à fait analogue (voir [3, Partie II, pp.45-49] pour la dérivation). Cette fois-ci, puisque

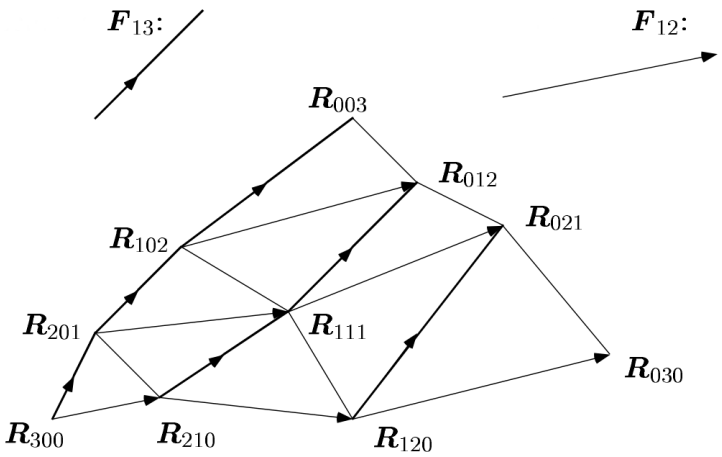

Figure 2.3 – Dessin tiré de [3] montrant les vecteurs de différence.

nous avons deux surfaces, l'espace paramétrique dans lequel est construite la fonction est quadridimensionnel et nous avons six ensembles de test-point plutôt que trois. Les ensembles de test-point, réutilisant la notation de [2, pp.519, 520], sont :

$$\begin{aligned}
\boldsymbol{Q}_a &= \{\boldsymbol{R}^a_{ijklm} = \boldsymbol{R}^1_{i+j,m,k+l} - \boldsymbol{R}^0_{i,j+l+m,k}\},\\
\boldsymbol{Q}_b &= \{\boldsymbol{R}^b_{ijklm} = \boldsymbol{R}^1_{j+l+m,i,k} - \boldsymbol{R}^0_{m,i+j,k+l}\},\\
\boldsymbol{Q}_c &= \{\boldsymbol{R}^c_{ijklm} = \boldsymbol{R}^1_{i+j,k+l,m} - \boldsymbol{R}^0_{i,k,j+l+m}\},\\
\boldsymbol{Q}_d &= \{\boldsymbol{R}^d_{ijklm} = \boldsymbol{R}^1_{i,j+l+m,k} - \boldsymbol{R}^0_{i+j,m,k+l}\},\\
\boldsymbol{Q}_e &= \{\boldsymbol{R}^e_{ijklm} = \boldsymbol{R}^1_{m,i+j,k+l} - \boldsymbol{R}^0_{j+l+m,i,k}\},\\
\boldsymbol{Q}_f &= \{\boldsymbol{R}^f_{ijklm} = \boldsymbol{R}^1_{i,k,j+l+m} - \boldsymbol{R}^0_{i+j,k+l,m}\}
\end{aligned} \tag{2.13}$$

où $\boldsymbol{R}^1$ et $\boldsymbol{R}^0$ sont les deux ensembles de points de contrôle originaux et $i+j+k+l+m = n$. Le lecteur intéressé peut trouver la liste des 35 vecteurs formant l'ensemble $\boldsymbol{Q}_a$ pour le cas $n = 3$ à l'annexe I. Il trouvera peut-être cette dernière utile pour visualiser plus facilement l'origine de la construction de ces vecteurs et leur signification géométrique. Le développement des ensembles de vecteurs (2.13) peut être trouvé dans [2, sect. 3.2, pp.518-523], en prenant tous les vecteurs produits, même ceux qui sont exclus aux pages 521 et subséquentes, puisque les restrictions de l'article original ne s'appliquent

pas ici. Elles s'appliqueront plutôt aux surfaces adjacentes et seront présentées à la section suivante aux équations (2.14) à (2.17). Ce développement a été omis puisqu'il diffère assez peu de celui déjà présenté pour l'auto-intersection et nous ne le jugions pas nécessaire à la compréhension de la méthode.

Une fois les ensembles de test-point construits, il ne reste qu'à vérifier que chacune des enveloppes convexes est séparable de l'origine pour vérifier qu'il n'y a pas d'intersection, comme précédemment et toujours au sens de condition suffisante seulement.

2.2.3 Objets adjacents

Pour tester l'intersection sur deux objets adjacents, *i.e.*, ayant un point de contrôle ou une courbe frontière en commun, deux méthodes ont été proposées dans [2]. La première opère comme pour l'intersection entre objets distincts et l'auto-intersection : elle construit des ensembles de test-point en calculant des différences de points de contrôle originaux. La deuxième, elle, construit des ensembles de test-point en faisant des différences de différences de points de contrôle. La deuxième est clairement plus lourde calculatoirement, puisqu'elle effectue plus de calcul pour en arriver à ses ensembles de test-point. Or, nous verrons à la section 3.2 que la première méthode, contrairement à ce que nous pourrions croire et, bien que plus simple, n'est pas moins puissante pour autant.

Commençons par présenter la *méthode principale*, soit la première mentionnée plus haut. Les ensembles de test-point utilisés sont les mêmes que ceux présentés à l'équation (2.13). Toutefois, ils ne peuvent pas être utilisés tels quels pour des surfaces adjacentes. Les différences entre les points communs aux deux surfaces doivent être enlevées, puisqu'elles seraient nécessairement égales à $\mathbf{0}$. Pour deux surfaces coïncidant le long de la courbe $r_0 = r_1 = 0$ en espace paramétrique, donc $\boldsymbol{R}^0_{0jk} = \boldsymbol{R}^1_{0jk}$ avec $j + k = n$, par exemple, les ensembles test seraient [2, p.521] :

$$\boldsymbol{Q}'_\nu = \{\boldsymbol{R}^\nu_{ijklm} \in \boldsymbol{Q}_\nu : k + m < n\}, \tag{2.14}$$

pour $\nu = a, c, d, f$, et

$$\boldsymbol{Q}'_\nu = \{\boldsymbol{R}^\nu_{ijklm} \in \boldsymbol{Q}_\nu : i + k < n\}, \tag{2.15}$$

pour $\nu = b, e$.

De la même façon, pour deux surfaces ayant un point de contrôle en commun, $r_0 = r_1 = 1$, soit $\boldsymbol{R}^0_{n00} = \boldsymbol{R}^1_{n00}$, par exemple, les ensembles test seraient [2, p.522] :

$$\boldsymbol{Q}'_\nu = \{\boldsymbol{R}^\nu_{ijklm} \in \boldsymbol{Q}_\nu : i < n\}, \tag{2.16}$$

pour $\nu = a, c, d, f$, et

$$\boldsymbol{Q}'_\nu = \{\boldsymbol{R}^\nu_{ijklm} \in \boldsymbol{Q}_\nu : m < n\}, \tag{2.17}$$

pour $\nu = b, e$.

À nouveau, une fois les ensembles de test-point construits, il ne reste qu'à vérifier que leurs enveloppes convexes sont séparables de l'origine pour vérifier qu'il n'y a pas d'intersection non désirée, au sens de condition suffisante seulement.

Méthode alternative

La *méthode alternative* pour l'intersection sur des objets adjacents discutée plus haut est construite comme des différences des points des ensembles de différences présentés en (2.13). La dérivation de ces ensembles est encore une fois assez semblable en essence à celle de la sous-section 2.2.1. Le lecteur intéressé peut se référer à [3, Partie II, pp.49-54] pour la dérivation détaillée. Reprenant la notation de [3, Partie II, p.45)], ces ensembles sont, pour deux surfaces coïncidant le long de la courbe $r_0 = r_1 = 0$ en espace

paramétrique :

$$
\begin{aligned}
\boldsymbol{F}^a_{51} &= \{C(1,0,0,0,-1)\boldsymbol{R}^c_{ijklm}\} = \{C(1,-1,0)\boldsymbol{R}^1_{i+j,m,k+l} + C(-1,1,0)\boldsymbol{R}^0_{i,j+l+m,k}\},\\
\boldsymbol{F}^a_{52} &= \{C(0,1,0,0,-1)\boldsymbol{R}^c_{ijklm}\} = \{C(1,-1,0)\boldsymbol{R}^1_{i+j,m,k+l}\},\\
\boldsymbol{F}^a_{54} &= \{C(0,0,0,1,-1)\boldsymbol{R}^c_{ijklm}\} = \{C(0,-1,1)\boldsymbol{R}^1_{i+j,m,k+l}\},\\
\\
\boldsymbol{F}^b_{12} &= \{C(-1,1,0,0,0)\boldsymbol{R}^c_{ijklm}\} = \{C(1,-1,0)\boldsymbol{R}^1_{j+l+m,i,k}\},\\
\boldsymbol{F}^b_{14} &= \{C(-1,0,0,1,0)\boldsymbol{R}^c_{ijklm}\} = \{C(1,-1,0)\boldsymbol{R}^1_{j+l+m,i,k} + C(0,1,-1)\boldsymbol{R}^0_{m,i+j,k+l}\},\\
\boldsymbol{F}^b_{15} &= \{C(-1,0,0,0,1)\boldsymbol{R}^c_{ijklm}\} = \{C(1,-1,0)\boldsymbol{R}^1_{j+l+m,i,k} + C(-1,1,0)\boldsymbol{R}^0_{m,i+j,k+l}\},\\
\\
\boldsymbol{F}^c_{51} &= \{C(1,0,0,0,-1)\boldsymbol{R}^c_{ijklm}\} = \{C(1,0,-1)\boldsymbol{R}^1_{i+j,k+l,m} + C(-1,0,1)\boldsymbol{R}^0_{i,k,j+l+m}\},\\
\boldsymbol{F}^c_{52} &= \{C(0,1,0,0,-1)\boldsymbol{R}^c_{ijklm}\} = \{C(1,0,-1)\boldsymbol{R}^1_{i+j,k+l,m}\},\\
\boldsymbol{F}^c_{54} &= \{C(0,0,0,1,-1)\boldsymbol{R}^c_{ijklm}\} = \{C(0,1,-1)\boldsymbol{R}^1_{i+j,k+l,m}\},
\end{aligned}
\tag{2.18}
$$

où $i+j+k+l+m = n-1$ et

$$
\begin{aligned}
\boldsymbol{q}_a &= \boldsymbol{F}^a_{51} \cup \boldsymbol{F}^a_{52} \cup \boldsymbol{F}^a_{54},\\
\boldsymbol{q}_b &= \boldsymbol{F}^b_{12} \cup \boldsymbol{F}^b_{14} \cup \boldsymbol{F}^b_{15},\\
\boldsymbol{q}_c &= \boldsymbol{F}^c_{51} \cup \boldsymbol{F}^c_{52} \cup \boldsymbol{F}^c_{54}.
\end{aligned}
\tag{2.19}
$$

Les ensembles $\boldsymbol{F}^\nu$ pour $\nu = d, e, f$ peuvent être obtenus en prenant les ensembles $\boldsymbol{F}^\nu$ pour $\nu = a, b, c$ respectivement, en inversant les indices 0 et 1 sur $\boldsymbol{R}$ et en changeant le signe de l'opérateur de de Casteljau. Par exemple, pour $\boldsymbol{F}^d_{51}$ nous aurions $\{C(-1,1,0)\boldsymbol{R}^0_{i+j,m,k+l} + C(1,-1,0)\boldsymbol{R}^1_{i,j+l+m,k}\}$ et pour $\boldsymbol{F}^e_{14}$, $\{C(-1,1,0)\boldsymbol{R}^0_{j+l+m,i,k} + C(0,-1,1)\boldsymbol{R}^1_{m,i+j,k+l}\}$. Les ensembles $\boldsymbol{q}_\nu,\ \nu = d, e, f$ sont définis similairement sur les unions des $\boldsymbol{F}^\nu$ correspondants.

De même et comme présenté dans [3, Partie II, pp.48-49)], pour deux surfaces ayant

un point de contrôle en commun, $r_0 = r_1 = 1$, nous obtenons les ensembles :

$$\begin{aligned}
\boldsymbol{F}^a_{12} &= \{C(1,-1,0)\boldsymbol{R}^0_{i,j+l+m,k}\}\\
\boldsymbol{F}^a_{13} &= \{C(-1,0,1)\boldsymbol{R}^1_{i+j,m,k+l} + C(1,0,-1)\boldsymbol{R}^0_{i,j+l+m,k}\}\\
\boldsymbol{F}^a_{14} &= \{C(-1,0,1)\boldsymbol{R}^1_{i+j,m,k+l} + C(1,-1,0)\boldsymbol{R}^0_{i,j+l+m,k}\}\\
\boldsymbol{F}^a_{15} &= \{C(-1,1,0)\boldsymbol{R}^1_{i+j,m,k+l} + C(1,-1,0)\boldsymbol{R}^0_{i,j+l+m,k}\},
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{F}^b_{51} &= \{C(-1,1,0)\boldsymbol{R}^1_{j+l+m,i,k} + C(1,-1,0)\boldsymbol{R}^0_{m,i+j,k+l}\}\\
\boldsymbol{F}^b_{52} &= \{C(1,-1,0)\boldsymbol{R}^0_{m,i+j,k+l}\}\\
\boldsymbol{F}^b_{53} &= \{C(-1,0,1)\boldsymbol{R}^1_{j+l+m,i,k} + C(1,0,-1)\boldsymbol{R}^0_{m,i+j,k+l}\}\\
\boldsymbol{F}^b_{54} &= \{C(1,0,-1)\boldsymbol{R}^0_{m,i+j,k+l}\},
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{F}^c_{12} &= \{C(1,0,-1)\boldsymbol{R}^0_{i,k,j+l+m}\}\\
\boldsymbol{F}^c_{13} &= \{C(-1,1,0)\boldsymbol{R}^1_{i+j,k+l,m} + C(1,-1,0)\boldsymbol{R}^0_{i,k,j+l+m}\}\\
\boldsymbol{F}^c_{14} &= \{C(-1,1,0)\boldsymbol{R}^1_{i+j,k+l,m} + C(1,0,-1)\boldsymbol{R}^0_{i,k,j+l+m}\}\\
\boldsymbol{F}^c_{15} &= \{C(-1,0,1)\boldsymbol{R}^1_{i+j,k+l,m} + C(1,0,-1)\boldsymbol{R}^0_{i,k,j+l+m}\},
\end{aligned}$$

où $i + j + k + l + m = n - 1$ et

$$\begin{aligned}
\boldsymbol{q}_a &= \boldsymbol{F}^a_{12} \cup \boldsymbol{F}^a_{13} \cup \boldsymbol{F}^a_{14} \cup \boldsymbol{F}^a_{15}\\
\boldsymbol{q}_b &= \boldsymbol{F}^b_{51} \cup \boldsymbol{F}^b_{52} \cup \boldsymbol{F}^b_{53} \cup \boldsymbol{F}^b_{54}\\
\boldsymbol{q}_c &= \boldsymbol{F}^c_{12} \cup \boldsymbol{F}^c_{13} \cup \boldsymbol{F}^c_{14} \cup \boldsymbol{F}^c_{15}.
\end{aligned}$$

Encore une fois, les ensembles $\boldsymbol{F}_\nu \mid \nu = d, e, f$ peuvent être obtenus à partir des ensembles $\boldsymbol{F}_a$, $\boldsymbol{F}_b$ et $\boldsymbol{F}_c$ par changement de signe et d'indice. Puis, les $\boldsymbol{q}_\nu$ sont obtenus en prenant l'union des $\boldsymbol{F}_\nu$ correspondants.

Comme pour la méthode principale, après avoir construit les ensembles, il ne reste qu'à vérifier qu'ils sont séparables de l'origine pour s'assurer de l'absence d'intersection non désirée, encore au sens de condition suffisante seulement.

2.3 Méthode Volino-Thalmann

La méthode de détection d'auto-intersection telle qu'initialement présentée par Volino et Thalmann [33] visait les maillages polygonaux. Elle a par la suite été transcrite aux surfaces paramétriques [5]. Dans ce mémoire, nous étudierons principalement la méthode Volino-Thalmann dans sa forme paramétrique pour la comparer à [3], mais commençons tout de même par regarder sa formulation originale.

L'idée derrière la méthode est très intuitive. En effet, pour s'assurer qu'il n'y a pas d'auto-intersection sur une surface, il suffit de vérifier que deux conditions sont respectées :

- Il existe un vecteur $\boldsymbol{v}$ pour lequel $\boldsymbol{n} \cdot \boldsymbol{v} > 0$ pour (presque) tous les points de la surface, où $\boldsymbol{n}$ est le vecteur normal au point en question. (Voir la figure 2.4 et la figure 2.5(a)).

- Le contour de la projection sur le plan perpendiculaire à $\boldsymbol{v}$ ne s'auto-intersecte pas (Voir la figure 2.5(b)).

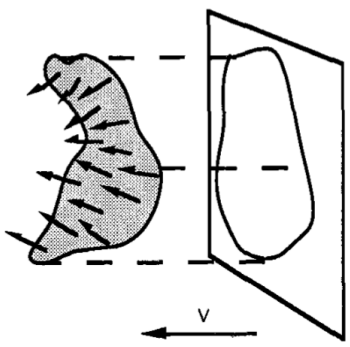

Figure 2.4 – Dessin explicatif tiré de l'article original [33].

Ce test est à première vue d'une très grande simplicité et seul le problème de trouver $\boldsymbol{v}$ peut sembler plus ardu. La technique proposée dans [33] pour réussir à trouver un tel vecteur est de faire un échantillonnage uniforme sur une sphère. Pour chacun des vecteurs normaux de la surface, on teste pour chacun des vecteurs de l'échantillon si

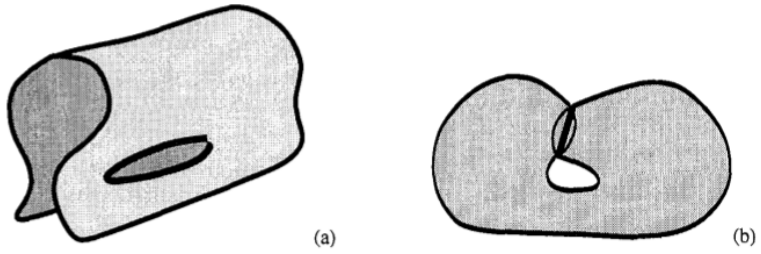

Figure 2.5 – Dessin explicatif tiré de l'article original [33].

leur produit scalaire est positif, voir la figure 2.6. On garde tous les vecteurs possibles à chaque point de la surface et on prend l'intersection des ensembles de vecteurs possibles pour tous les points de la surface pour trouver un vecteur $\boldsymbol{v}$ valide pour l'ensemble de la surface. Si l'intersection des ensembles est vide, alors la première condition n'est pas satisfaite et le test ne peut pas garantir la non-auto-intersection.

Contrairement aux méthodes vues jusqu'à maintenant, on voit que celle-ci a une puissance et un temps d'exécution variable. L'utilisateur, en choisissant un nombre d'échantillons plus ou moins grand, peut avoir plus ou moins de précision au coût d'un temps de calcul plus ou moins grand. La précision est variable, mais tout de même bornée. La précision maximale serait atteinte théoriquement en prenant un nombre d'échantillons tendant vers l'infini. Or, même avec un nombre infini d'échantillons, la normale devrait tout de même être circonscrite dans un cône de demi-angle plus

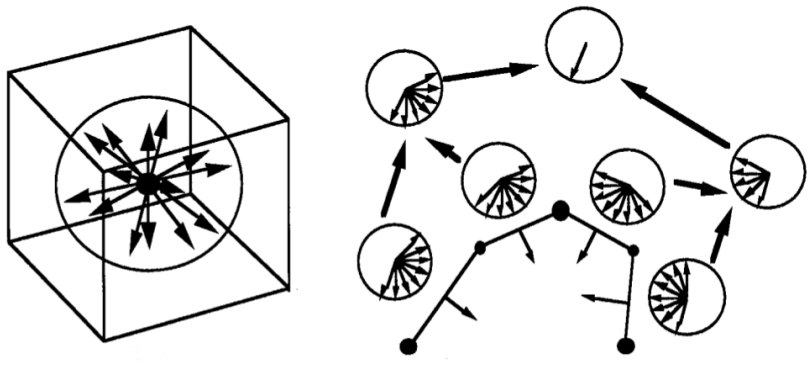

Figure 2.6 – Dessin explicatif tiré de l'article original [33].

petit que $\frac{\pi}{2}$. Cette méthode demeure donc, comme celles vues jusqu'à maintenant, une condition suffisante seulement. On peut voir que même sur des surfaces simples, comme celle présentée à la figure 3.10, le test ne réussit pas à garantir la non-auto-intersection.

L'article original traite de surfaces polygonales, mais ce n'est pas ce qui nous intéresse ici. Nous allons plutôt étudier la version paramétrique de la méthode Volino-Thalmann, telle que décrite dans [5]. Pour les surfaces paramétriques, les conditions doivent être précisées. En effet, contrairement à ce qui est fait dans l'article original, nous ne pouvons pas simplement faire un échantillonnage de la normale. Il faudra maintenant réussir à borner la normale sur un intervalle et vérifier que celui-ci peut être inclus dans un cône de demi-angle inférieur ou égal à $\frac{\pi}{2}$, ce qui demandera passablement plus de travail. Or, même borner la normale dans un cône ne sera pas non plus suffisant, il faudra aussi s'assurer explicitement que la normale n'est égale au vecteur nul en aucun point. Ce cas est le cas limite d'une intersection et doit être vérifié explicitement sans quoi la méthode échoue. Nous reviendrons sur la difficulté de borner la normale et la nécessité de vérifier qu'elle ne s'annule pas à la sous-section 4.2.1. Nous verrons que cette particularité est un désavantage important de la méthode. De plus, comme discuté à la section 3.5 de [5], le mot « presque » de la condition 1 prend alors le sens que $\boldsymbol{n} \cdot \boldsymbol{v} > 0$ seulement à l'intérieur du domaine, tout en ayant qu'il soit possible que $\boldsymbol{n} \cdot \boldsymbol{v} = 0$ sur la frontière du domaine. Finalement, la dernière condition sera que les surfaces étudiées aient une continuité au moins C^1 et que leur contour soit une courbe C^1-continue par morceaux.

L'article [5] ne fait pas que formaliser les conditions de [33]. Il propose aussi une version de la méthode qui soit applicable aux surfaces composées, donne des contre-exemples montrant que toutes les conditions sont nécessaires et montre que la méthode ne peut pas fonctionner comme telle sur les domaines non simplement connexes. Pour la rendre applicable au domaine non simplement connexe, il faut aussi vérifier certaines conditions sur l'orientation des courbes formant les contours, ajoutant ainsi une autre condition préalable à vérifier. Dans ce mémoire nous étudierons seulement des cas de surfaces simplement connexes, mais il faut tout de même garder cette restriction à l'esprit si nous voulons éventuellement généraliser les résultats que nous avons obtenus. Les

résultats de [5] présentent un grand intérêt pratique. Plus particulièrement, le résultat pour les surfaces composées est très utile, puisque dans les contextes réels, les méthodes utilisées font souvent appel à une structure hiérarchique, citons [18] par exemple. Dans ces cas, il peut être utile et potentiellement plus efficace de pouvoir traiter plusieurs sous-surfaces à la fois. Or, dans cette version plus générale de la méthode, certaines conditions supplémentaires s'ajoutent. D'abord, mentionnons que la surface composée n'a pas besoin d'être C^1-continue comme précédemment, mais il faut seulement que chacune de ses sous-surfaces le soit et de même pour leurs contours. Elles devront aussi avoir un domaine paramétrique commun. Ensuite, il faudra, non seulement que $\boldsymbol{n} \cdot \boldsymbol{v} > 0$ et que le contour de la surface ne s'intersecte pas, mais aussi que ces deux conditions soient respectées pour chacune des sous-surfaces. Finalement, des conditions de régularité devront aussi être vérifiées à la jonction entre les sous-surfaces. Bref, cette version plus générale est aussi passablement plus complexe à implémenter. Le lecteur intéressé peut trouver tous les détails de la méthode Volino-Thalmann paramétrique dans [5, sect. 4], sur les différentes conditions préalables dans [5, sect. 2.1, 3] et les contre-exemples montrant que les conditions sont nécessaires dans [5, sect. 5].

2.4 Structures de haut niveau

Dans des applications réelles, les surfaces sur lesquelles nous souhaitons détecter des problèmes potentiels sont généralement composées, au sens où elles sont constituées de plusieurs sous-surfaces assemblées pour former le modèle complet. Dans ces cas, le travail de détection n'est pas fait sur le modèle entier, mais plutôt sur les sous-surfaces, puisqu'il est possible d'utiliser des structures d'accélération pour éviter autant que possible les tests individuels inutiles. Les sous-surfaces sont alors testées pour l'auto-intersection et pour l'intersection entre elles deux à deux.

2.4.1 Hiérarchisation

Les algorithmes de haut niveau qui ont été proposés pour détecter des intersections non désirées sur des objets géométriques complexes font quasi toujours appel à une

certaine forme de hiérarchisation, puisqu'elle est relativement simple d'implémentation et permet d'exclure rapidement beaucoup de tests inutiles. Ces méthodes peuvent être catégorisées en deux familles principales. Une première famille d'approches est basée sur le groupement spatial. Nous pourrions citer ici, par exemple, les *Bounding Volume Hierarchy* (BVH), les *Binary Space Partition Tree* (BSP-Tree) et les *k-d tree*. L'idée derrière ces méthodes repose sur le fait que deux surfaces qui sont éloignées ne peuvent pas s'intersecter. Ces méthodes permettent, au coût d'un précalcul pour la construction de la structure, de réduire considérablement le nombre de tests individuels. Elles permettent en moyenne de faire passer le nombre de tests individuels à $\Theta(n \lg n)$, ce qui, en procédant naïvement aurait été de $\Theta(n^2)$, où n est le nombre de sous-surfaces. Sur des maillages de grande taille, l'avantage est grand, même en tenant compte du coût de fabrication de la structure, qui est lui aussi de $\Theta(n \lg n)$. Ces approches sont très efficaces, mais ont tout de même une puissance limitée. Elles ne peuvent opérer que sur des ensembles de sous-surfaces disjoints. Elles ne sont donc pas utilisables pour exclure les tests sur des sous-surfaces adjacentes. Cet inconvénient pourrait être partiellement mitigé par l'utilisation de la deuxième famille d'approches. Cette deuxième famille est, elle, basée sur l'analyse du vecteur normal à la surface. L'idée cette fois est que pour un groupe de sous-surfaces adjacentes, il ne peut y avoir d'intersection si le vecteur normal à la surface ne varie pas beaucoup. Il suffira donc de pouvoir borner le vecteur normal dans un intervalle pour pouvoir assurer la non-intersection. C'est dans cette famille que nous pourrions classer la méthode présentée à la section 2.3, ainsi que sa généralisation à des surfaces paramétriques [5] ou sa généralisation aux surfaces de subdivision de Loop [18]. Les deux familles de méthodes sont essentiellement différentes. En effet, la première opére uniquement sur des surfaces disjointes, alors que la deuxième opère uniquement sur des surfaces adjacentes. Les méthodes basées sur la distance sont utilisées presque partout et peuvent aisément être combinées aux méthodes d'étude de la normale. Grinspun et Schroeder, par exemple, dans [18], donnent un bel exemple d'algorithme combinant les deux idées pour tenter de réduire au maximum les tests individuels. Les deux familles sont réellement complémentaires. Il ne servirait à rien de faire un test comme AABB sur des surfaces adjacentes, il échouerait assurément. Il ne serait pas non

plus utile de tester pour la normale entre deux objets disjoints puisque cela ne nous apprendrait rien sur la possibilité d'intersection. Finalement, notons que l'adjacence n'est pas uniquement définie entre les sous-surfaces d'un même niveau hiérarchique, mais aussi entre les niveaux. Ainsi, il sera possible de tester pour l'intersection entre deux sous-surfaces adjacentes de niveaux différents. Plus de détails sur ces idées de hiérarchie et d'adjacence peuvent être trouvés dans [18, sect. 2], par exemple.

Ces approches, tant celles basées sur la distance que celles basées sur la normale, sont générales et applicables sur toutes les surfaces, peu importe leur degré. Toutefois, pour les méthodes de groupement spatial, afin que ce soit valide pour les surfaces paramétriques, il faudra avoir la propriété de l'enveloppe convexe. De même, pour les méthodes d'étude de la normale, il faudra être en mesure de pouvoir borner cette dernière. Pour toutes les surfaces les plus utilisées, Bézier, B-splines ou NURBS, par exemple, nous avons ces propriétés, ce ne sera donc pas un problème. Or, il faut tout de même garder en tête que le problème de trouver une borne précise sur la normale est un problème difficile et a un coût calculatoire important.

2.4.2 Subdivision

En plus de l'organisation hiérarchique des surfaces, une deuxième idée est utilisée dans la plupart des algorithmes de haut niveau : la subdivision. Celle-ci et la hiérarchisation vont de paire puisque, à chaque étape de subdivision, la nouvelle surface créée peut être vue comme le niveau hiérarchique inférieur de la surface originale. L'idée derrière ces approches est de subdiviser la surface en plusieurs sous-surfaces dont l'union équivaut à la surface originale et d'appliquer le test de détection d'intersection sur les sous-surfaces plutôt que sur la surface originale. Puisque chacune des sous-surfaces est plus petite et moins courbée que la surface originale, le test après subdivision n'est jamais plus faible que le test sur la surface originale, comme nous le verrons à la section 3.4, d'où son intérêt.

Les surfaces de Bézier, grâce à l'algorithme de de Casteljau, peuvent être subdivisées en un ou plusieurs points arbitraires. Les deux méthodes les plus utilisées pour subdiviser sont la méthode originale de de Casteljau, décrite par Farin dans l'article [12] et la

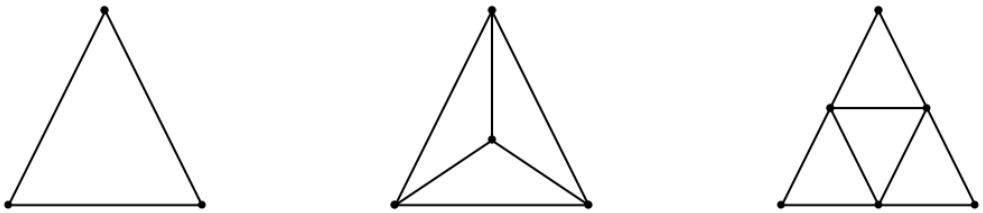

Figure 2.7 – Dessin des principaux schémas de subdivision. À gauche : l'espace paramétrique original ; au centre : l'algorithme de Farin ; à droite : l'algorithme de Goldman.

généralisation de l'idée de de Casteljau, utilisant des simplexes de dimension plus élevée, présentée par Goldman dans [15]. La méthode de Farin, plus simple et moins coûteuse en calcul, permet de subdiviser la surface originale en trois sous-surfaces selon un point arbitraire (la figure 2.7, au centre). La méthode de Goldman, quant à elle, est un peu plus complexe, mais permet de subdiviser en quatre sous-surfaces selon trois points arbitraires. En fait, l'article de Goldman présente plusieurs méthodes différentes permettant de subdiviser la surface dans un nombre allant jusqu'à sept sous-surfaces. Dans ce mémoire nous n'accorderons d'attention qu'à la version de subdivision par quatre puisque ces résultats sont bien plus intéressants, et c'est de loin la plus utilisée. Les autres méthodes de l'article donnent des résultats qui, comme pour la méthode de Farin, sont moins stables. En effet, comme nous pouvons le voir à la figure 2.7, les sous-surfaces que la subdivision par quatre génère ont des ratios beaucoup plus équilibrés que la méthode de Farin. Cette caractéristique sera très importante lorsque nous subdiviserons puisque les sous-surfaces générées seront ainsi moins sensibles aux erreurs numériques. Nous discuterons plus en détail notre implémentation à la section 3.4.

CHAPITRE 3

ORDRE PARTIEL ENTRE LES MÉTHODES (PUISSANCE)

Commençons par comparer les méthodes d'un point de vue théorique, selon leur puissance de discrimination. Nous nous intéresserons ici seulement à la puissance, sans égard au coût. Ce deuxième critère sera étudié plus tard.

3.1 Comparaison entre la méthode test-point et la séparation d'enveloppes convexes

Nous verrons dans la section qui suit que la méthode test-point pour les objets distincts (sous-section 2.2.2) est strictement plus puissante que la méthode de la séparation des enveloppes convexes (section 2.1), et ce, tant pour les courbes que les surfaces. Pour ce faire, il suffit de montrer deux choses : que la méthode test-point peut toujours garantir la non-intersection lorsque la méthode de l'enveloppe convexe peut le faire, puis qu'il existe des cas où la méthode test-point est concluante, mais où la méthode standard de séparation des enveloppes ne l'est pas.

3.1.1 La méthode test-point n'est jamais moins puissante

La preuve qui suit s'applique à la méthode test-point pour les surfaces, mais une dérivation tout à fait analogue et plus simple peut être faite pour la méthode opérant sur les courbes. Remarquons d'abord que tous les vecteurs $\boldsymbol{R}^x_{ijklm}$ de (2.13) ont la forme $\boldsymbol{R}^1_{i_1j_1k_1} - \boldsymbol{R}^0_{i_0j_0k_0}$. Ainsi, si nous avons un vecteur $\boldsymbol{n}$ séparant les enveloppes convexes des deux ensembles de points de contrôle, alors nous avons aussi que $\min_{i_1j_1k_1}\{\boldsymbol{n} \cdot \boldsymbol{R}^1_{i_1j_1k_1}\} > \max_{i_0j_0k_0}\{\boldsymbol{n} \cdot \boldsymbol{R}^0_{i_0j_0k_0}\}$, ce qui implique que $\boldsymbol{n} \cdot \boldsymbol{R}^x_{ijklm} > 0$. Le même vecteur $\boldsymbol{n}$ séparant les deux enveloppes convexes peut donc aussi séparer l'enveloppe convexe des test-point de l'origine et ceci pour tous les $x = a, b, c, d, e, f$.

Une autre approche permettant de voir que la méthode test-point ne peut pas échouer lorsque la méthode de séparation des enveloppes convexes fonctionne serait de montrer que s'il existe des constantes non négatives λ^x_{ijklm} qui somment à 1 telles que

$\sum_{ijklm} \lambda^x_{ijklm} \boldsymbol{R}^x_{ijklm} = \mathbf{0}$, alors il existe aussi des constantes non négatives μ^x_{ijk} et ν^x_{ijk} qui somment toutes deux à 1 telles que

$$\sum_{ijk} \mu^x_{ijk} \boldsymbol{R}^1_{ijk} = \sum_{ijk} \nu^x_{ijk} \boldsymbol{R}^0_{ijk}. \tag{3.1}$$

Si de telles constantes existent, elles prouveraient que lorsque la méthode test-point échoue alors la méthode de séparation des enveloppes convexes échoue aussi. Par la contraposée, nous obtiendrions effectivement le même résultat que précédemment, soit que si la méthode de séparation des enveloppes réussit, alors la méthode test-point réussit aussi, donc qu'elle n'est jamais plus faible. Or ces constantes existent. Pour le cas a par exemple, nous avons que

$$\boldsymbol{R}^a_{i'j'k'l'm'} = \boldsymbol{R}^1_{i'+j',m',k'+l'} - \boldsymbol{R}^0_{i',j'+l'+m',k'}, \tag{3.2}$$

en utilisant les indices i', j', k', l', m', plutôt que les i, j, k, l, m que nous avions précédemment à l'équation (2.13). Si nous multiplions cette équation par $\lambda^a_{i'j'k'l'm'}$ et sommons sur tous les indices, nous voyons que pour que la condition (3.1) soit satisfaite, il suffit de s'assurer que les ν^a_{ijk} soient égaux à la somme des $\lambda^a_{i'j'k'l'm'}$ pour lesquels l'indice de $\boldsymbol{R}^0$ est égal à (i, j, k) et de même pour les μ^a_{ijk}. Ceci peut être fait en prenant

$$\nu^a_{ijk} = \sum_{j'=0}^{j} \sum_{l'=0}^{j-j'} \lambda^a_{i,j',k,l',j-j'-l'} \tag{3.3}$$

et

$$\mu^a_{ijk} = \sum_{i'=0}^{i} \sum_{k'=0}^{k} \lambda^a_{i',i-i',k',k-k',j}. \tag{3.4}$$

Le lecteur intéressé peut trouver la liste des coefficients ν et μ pour $n = 3$ à l'annexe II.

Des expressions semblables peuvent être trouvées pour $x = b, c, d, e, f$. Nous voyons donc à nouveau que la méthode test-point n'est jamais plus faible que la méthode de séparation des enveloppes convexes.

3.1.2 La méthode test-point est parfois plus puissante

Commençons par donner un exemple de courbes de Bézier dans le plan pour lesquelles la méthode test-point est concluante, mais la séparation de l'enveloppe convexe ne l'est pas. Prenons les deux courbes de Bézier définies par les ensembles de test-point suivants :

$$\begin{array}{ccc} \text{(-1,1)} & \text{(1,1)} & \text{(1,-1)} \\ \boldsymbol{R}_0^1 & \boldsymbol{R}_1^1 & \boldsymbol{R}_2^1 \end{array} \tag{3.5}$$

et

$$\begin{array}{ccc} \text{(-2,0)} & \text{(0,0)} & \text{(0,-2)} \\ \boldsymbol{R}_0^0 & \boldsymbol{R}_1^0 & \boldsymbol{R}_2^0 \end{array} \tag{3.6}$$

Leurs enveloppes convexes sont inséparables puisque le point $(0, 0)$ appartient aux deux, comme nous pouvons le voir à la figure 3.1. Or, les deux ensembles de test-point pour

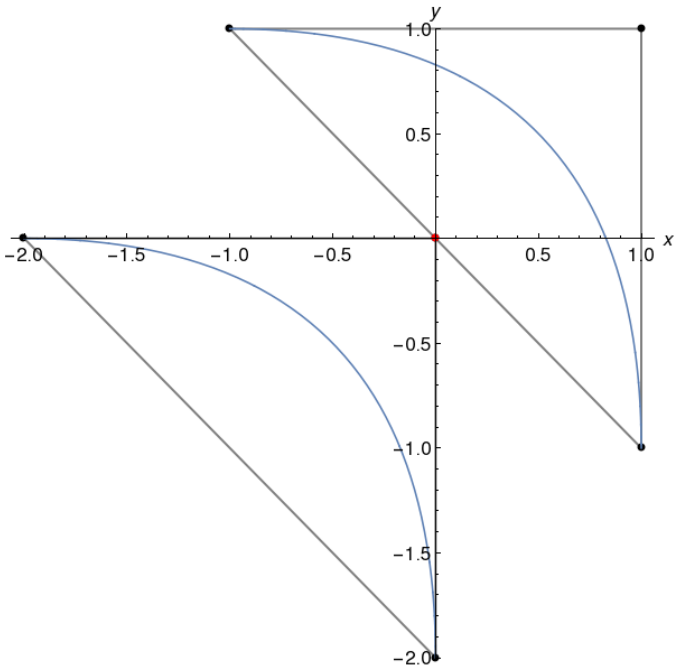

Figure 3.1 – Enveloppes convexes des points de contrôle des deux courbes de Bézier générées par les points de contrôle (3.5) et (3.6).

tester l'intersection, comme donné dans [2, (14), (15), p.513] et à la section 2.2.2, sont :

$$\begin{matrix} \boldsymbol{R}_2^1 - \boldsymbol{R}_2^0 & \boldsymbol{R}_1^1 - \boldsymbol{R}_2^0 & \boldsymbol{R}_0^1 - \boldsymbol{R}_2^0 \\ \boldsymbol{R}_1^1 - \boldsymbol{R}_1^0 & \boldsymbol{R}_0^1 - \boldsymbol{R}_1^0 & \\ \boldsymbol{R}_0^1 - \boldsymbol{R}_0^0 & & \end{matrix} \tag{3.7}$$

pour $\boldsymbol{Q}_a$ et

$$\begin{matrix} \boldsymbol{R}_2^1 - \boldsymbol{R}_2^0 & \boldsymbol{R}_2^1 - \boldsymbol{R}_1^0 & \boldsymbol{R}_2^1 - \boldsymbol{R}_0^0 \\ \boldsymbol{R}_1^1 - \boldsymbol{R}_1^0 & \boldsymbol{R}_1^1 - \boldsymbol{R}_0^0 & \\ \boldsymbol{R}_0^1 - \boldsymbol{R}_0^0 & & \end{matrix} \tag{3.8}$$

pour $\boldsymbol{Q}_b$. Nous avons donc :

$$\begin{aligned} \boldsymbol{Q}_a &= \{(1,1),(1,3),(-1,3),(-1,1)\}, \\ \boldsymbol{Q}_b &= \{(1,1),(1,-1),(3,-1),(3,1)\}, \end{aligned} \tag{3.9}$$

qui sont tous les deux séparables de l'origine, comme on peut le voir sur la figure 3.2 et la figure 3.3.

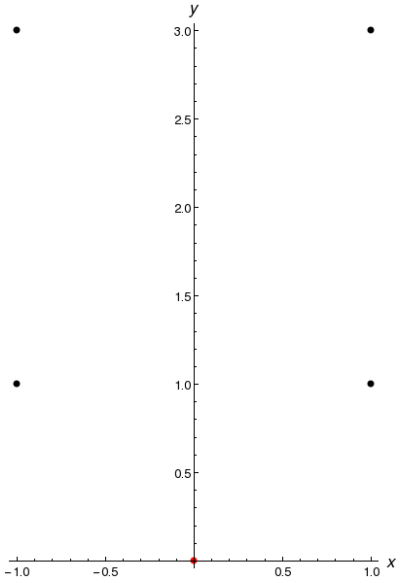

Figure 3.2 – Test-point de $\boldsymbol{Q}_a$.

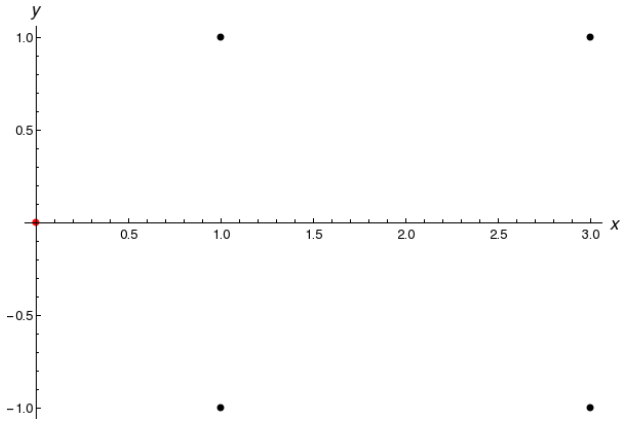

Figure 3.3 – Test-point de $\boldsymbol{Q}_b$.

Ainsi, nous voyons qu'il existe des cas où la méthode test-point réussit, mais que la méthode standard échoue. De plus, il est intéressant de noter que cet exemple n'est pas unique ou insolite. Nous l'avons choisi puisque c'est le cas limite et qu'il présente, pour cette raison, un intérêt supplémentaire, n'ayant qu'un unique point commun aux deux enveloppes. Or, nous aurions aussi pu choisir un exemple plus flagrant, en déplaçant les points de contrôle de la deuxième courbe vers les x et les y positifs. Les deux courbes de la figure 3.4 présentent un tel cas où les enveloppes se chevauchent sur une beaucoup plus grande surface et où le résultat des deux tests est le même. En fait, nous aurions pu choisir n'importe quel ensemble de points de contrôle ayant la forme

$$\underset{\boldsymbol{R}_0^0}{\big(-(1+m),1-m\big)} \quad \underset{\boldsymbol{R}_1^0}{\big(1-m,1-m\big)} \quad \underset{\boldsymbol{R}_2^0}{\big(1-m,-(1+m)\big)} \;, \tag{3.10}$$

où $m \in \,]0,1]$ et aurions obtenu le même résultat. Pour toutes les valeurs de $m < 1$, les enveloppes se seraient chevauchées sur plus qu'un unique point. Nous voyons ici que la distance séparant les deux courbes aurait pu être infinitésimale et nous aurions toujours obtenu le même résultat des deux méthodes. Cet exemple montre donc bien une différence de puissance véritable entre les deux méthodes.

Notons surtout que le même constat peut être fait pour les surfaces : il suffit de prendre une extrusion des deux courbes données plus haut pour obtenir un exemple de

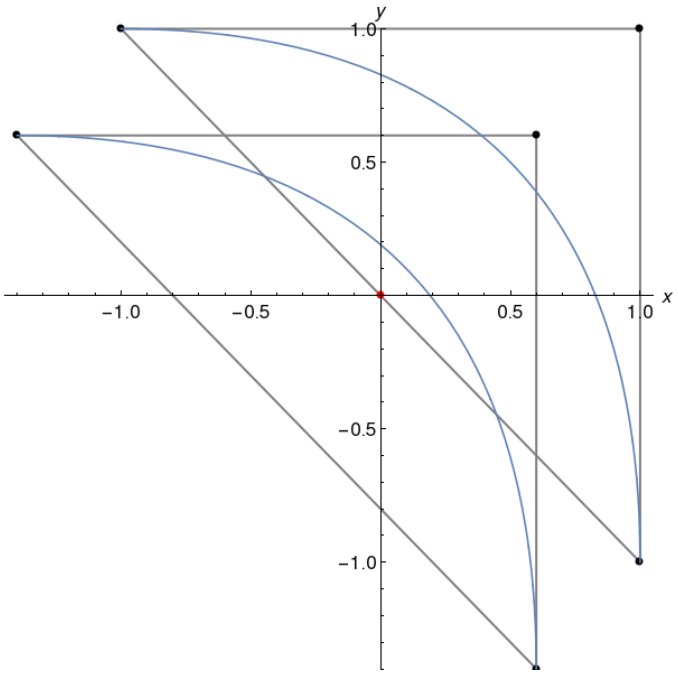

Figure 3.4 – Enveloppes convexes des points de contrôle des deux courbes de Bézier générées par les points de contrôle (3.5) et (3.6), mais où la deuxième courbe a été déplacée de 0,6 en x et en y positif.

deux surfaces qui ne sont pas séparables utilisant la méthode standard, mais qui le sont avec la méthode test-point. Nous avons choisi de montrer l'exemple pour les courbes ici puisqu'il était plus simple à visualiser.

Considérant les résultats des sections 3.1.1 et 3.1.2, nous voyons que la méthode test-point est strictement plus puissante que la méthode standard de séparation des enveloppes convexes, puisqu'elle n'est jamais moins puissante et est parfois plus puissante.

3.2 Comparaison entre la méthode test-point pour objets adjacents et la méthode alternative de [2]

Dans cette section et comme précédemment mentionné, nous allons voir que les deux méthodes proposées dans [2] ne sont pas équivalentes : la méthode principale (*cf.*, p.16) est strictement plus puissante que la méthode alternative (*cf.*, p.17). Le raisonnement

derrière la preuve pour la méthode opérant sur les courbes et celle opérant sur la surface est en tout point analogue. Nous débuterons donc par la preuve pour la méthode pour courbes puisqu'elle est plus simple et permettra d'acquérir de la compréhension sous-jacente qui aidera à suivre plus facilement la preuve pour les surfaces.

3.2.1 Courbes

En utilisant la notation de [3, Partie I, p.19], les ensembles $\boldsymbol{q}_a$ et $\boldsymbol{q}_b$ de la méthode alternative sont définis comme $\boldsymbol{F}^a_{12} \cup \boldsymbol{F}^a_{13}$ et $\boldsymbol{F}^b_{12} \cup \boldsymbol{F}^b_{13}$ respectivement où

$$\begin{aligned}
\boldsymbol{F}^a_{12} &= \{-\boldsymbol{r}^0_{j+k}\}_{i+j+k=n-1}, \\
\boldsymbol{F}^a_{13} &= \{\boldsymbol{r}^1_k - \boldsymbol{r}^0_{j+k}\}_{i+j+k=n-1}, \\
\boldsymbol{F}^b_{12} &= \{-\boldsymbol{r}^1_{j+k}\}_{i+j+k=n-1}, \\
\boldsymbol{F}^b_{13} &= \{\boldsymbol{r}^1_{j+k} - \boldsymbol{r}^0_k\}_{i+j+k=n-1}.
\end{aligned} \tag{3.11}$$

Les ensembles $\boldsymbol{Q}'_a$ et $\boldsymbol{Q}'_b$ de la méthode principale, sont définis respectivement par

$$\begin{aligned}
\boldsymbol{Q}'_a &= \{\boldsymbol{R}^a_{ijk}\}_{i+j+k=n,\ 0\leq i<n} = \{\boldsymbol{R}^1_k - \boldsymbol{R}^0_{j+k}\}, \\
\boldsymbol{Q}'_b &= \{\boldsymbol{R}^b_{ijk}\}_{i+j+k=n,\ 0\leq i<n} = \{-\boldsymbol{R}^0_k + \boldsymbol{R}^1_{j+k}\}.
\end{aligned} \tag{3.12}$$

Notons que l'indiçage de ces ensembles est le même que pour les ensembles de (3.11) à l'exception que les trois indices varient maintenant de 0 à n plutôt que de 0 à $n-1$. Toutefois, la dernière ligne de l'ensemble, celle correspondant à $j = 0$ et $i = n$ et ne contenant qu'un élément $\boldsymbol{R}^\nu_{n00} = \boldsymbol{R}^1_0 - \boldsymbol{R}^0_0 = 0$, est omise, comme stipulé dans la définition de $\boldsymbol{Q}'_\nu$, $\nu = a,\ b$ [3, Partie I, p.20].

Les tests des deux méthodes pour les courbes sont analogues à ceux pour les surfaces présentés dans la sous-section 2.2.3, à la seule différence qu'ils opèrent sur les ensembles présentés en (3.11) et (3.12) plutôt que sur les ensembles présentés au chapitre 2. Ainsi, comme précédemment, le test correspond à construire des ensembles de test-point, prendre l'enveloppe convexe de ces ensembles, puis vérifier si ces enveloppes sont séparables de l'origine.

Afin de montrer que la méthode principale n'est jamais moins puissante que la méthode alternative, nous allons montrer que chaque élément de l'ensemble $\boldsymbol{Q}'_\nu$ peut être réécrit comme une somme d'éléments de l'ensemble $\boldsymbol{q}_\nu$ et ce pour $\nu = a, b$. Il s'ensuit que pour chaque cas, si $\boldsymbol{x} \in \boldsymbol{F}^\nu_{12} \cup \boldsymbol{F}^\nu_{13}$ implique $\boldsymbol{n_\nu} \cdot \boldsymbol{x} > 0$, alors nous savons aussi que $\boldsymbol{y} \in \boldsymbol{Q}'_\nu$ implique $\boldsymbol{n_\nu} \cdot \boldsymbol{y} > 0$. Or, puisque ceci est vrai pour $\nu = a, b$, nous pouvons donc conclure que si la méthode alternative peut exclure la possibilité d'intersections non désirées, alors la méthode principale le peut aussi. Elle n'est donc jamais moins puissante.

Commençons donc par réécrire les vecteurs de $\boldsymbol{Q}'_a$ comme une somme de vecteurs de $\boldsymbol{q}_a$. Les vecteurs de $\boldsymbol{Q}'_a$, $\boldsymbol{R}^1_k - \boldsymbol{R}^0_{j+k} = \boldsymbol{R}^1_{n-i-j} - \boldsymbol{R}^0_{n-i}$, pour $j \geq 1$, peuvent s'écrire comme :

$$\left\{\begin{array}{l} +\boldsymbol{R}^1_{n-i-j} - \boldsymbol{R}^1_{n-i-j-1} \\ +\boldsymbol{R}^1_{n-i-j-1} - \boldsymbol{R}^1_{n-i-j-2} \\ +... \\ +\boldsymbol{R}^1_1 - \boldsymbol{R}^1_0 \\ +\boldsymbol{R}^1_0 \end{array}\right\} - \left\{\begin{array}{l} +\boldsymbol{R}^0_{n-i} - \boldsymbol{R}^0_{n-i-1} \\ +\boldsymbol{R}^0_{n-i-1} - \boldsymbol{R}^0_{n-i-2} \\ +... \\ +\boldsymbol{R}^0_{j+1} - \boldsymbol{R}^0_j \\ +\boldsymbol{R}^0_j - \boldsymbol{R}^0_{j-1} \\ +\boldsymbol{R}^0_{j-1} - \boldsymbol{R}^0_{j-2} \\ +... \\ +\boldsymbol{R}^0_1 - \boldsymbol{R}^0_0 \\ +\boldsymbol{R}^0_0 \end{array}\right\}$$

soit, de manière plus concise

$$\sum_{k=0}^{n-i-j-1} \boldsymbol{r}^1_k - \sum_{k=0}^{n-i-j-1} \boldsymbol{r}^0_{j+k} + \sum_{k=0}^{j-1}(-\boldsymbol{r}^0_k) + \boldsymbol{R}^1_0 - \boldsymbol{R}^0_0.$$

En fusionnant les sommes et en éliminant $\boldsymbol{R}^1_0 - \boldsymbol{R}^0_0$ puisqu'il vaut $\boldsymbol{0}$, nous obtenons :

$$\sum_{k=0}^{n-i-j-1} (\boldsymbol{r}^1_k - \boldsymbol{r}^0_{j+k}) + \sum_{k=0}^{j-1}(-\boldsymbol{r}^0_k).$$

Le premier terme est une somme d'éléments de $\boldsymbol{F}_{13}^{a}$ et le second une somme d'éléments de $\boldsymbol{F}_{12}^{a}$. Nous avons donc bel et bien une représentation d'un vecteur de $\boldsymbol{Q}_{a}^{\prime}$ exprimé comme une somme des vecteurs de $\boldsymbol{q}_{a}$. Le résultat précédent tient aussi pour le cas où $j = 0$. Dans ce cas, la somme des éléments de $\boldsymbol{F}_{12}^{a}$ disparaît.

De la même façon, les vecteurs de $\boldsymbol{Q}_{b}^{\prime}$, $\boldsymbol{R}_{j+k}^{1} - \boldsymbol{R}_{k}^{0} = \boldsymbol{R}_{n-i}^{1} - \boldsymbol{R}_{n-i-j}^{0}$, peuvent être écrits comme

$$\left\{\begin{array}{l} +\boldsymbol{R}_{n-i}^{1} - \boldsymbol{R}_{n-i-1}^{1} \\ +\boldsymbol{R}_{n-i-1}^{1} - \boldsymbol{R}_{n-i-2}^{1} \\ +... \\ +\boldsymbol{R}_{j+1}^{1} - \boldsymbol{R}_{j}^{1} \\ +\boldsymbol{R}_{j}^{1} - \boldsymbol{R}_{j-1}^{1} \\ +\boldsymbol{R}_{j-1}^{1} - \boldsymbol{R}_{j-2}^{1} \\ +... \\ +\boldsymbol{R}_{1}^{1} - \boldsymbol{R}_{0}^{1} \\ +\boldsymbol{R}_{0}^{1} \end{array}\right\} - \left\{\begin{array}{l} +\boldsymbol{R}_{n-i-j}^{0} - \boldsymbol{R}_{n-i-j-1}^{0} \\ +\boldsymbol{R}_{n-i-j-1}^{0} - \boldsymbol{R}_{n-i-j-2}^{0} \\ +... \\ +\boldsymbol{R}_{1}^{0} - \boldsymbol{R}_{0}^{0} \\ +\boldsymbol{R}_{0}^{0} \end{array}\right\}$$

ou plus simplement

$$\sum_{k=0}^{n-i-j-1} \boldsymbol{r}_{j+k}^{1} - \sum_{k=0}^{n-i-j-1} \boldsymbol{r}_{k}^{0} + \sum_{k=0}^{j-1} \boldsymbol{r}_{k}^{1} + \boldsymbol{R}_{0}^{1} - \boldsymbol{R}_{0}^{0}.$$

Encore une fois, en réarrangeant les sommes et en éliminant $\boldsymbol{R}_{0}^{1} - \boldsymbol{R}_{0}^{0}$, nous obtenons :

$$\sum_{k=0}^{n-i-j-1} (\boldsymbol{r}_{j+k}^{1} - \boldsymbol{r}_{k}^{0}) + \sum_{k=0}^{j-1} \boldsymbol{r}_{k}^{1}.$$

Le premier terme est une somme d'éléments de $\boldsymbol{F}_{13}^{b}$ et le deuxième une somme d'éléments de $\boldsymbol{F}_{12}^{b}$.

Nous avons donc bel et bien le résultat recherché, soit que tous les vecteurs de $\boldsymbol{Q}_{\nu}^{\prime}$

peuvent s'écrire comme des vecteurs de $\boldsymbol{q}_\nu$. Ainsi, le test $\boldsymbol{Q}'_\nu$ ne peut jamais échouer à exclure la possibilité d'intersection non désirée si le test $\boldsymbol{q}_\nu$ réussit. Il n'est donc jamais moins puissant.

Maintenant pour prouver que la méthode principale est parfois plus puissante que la méthode alternative, il suffit de trouver un exemple où la première arrive à exclure l'intersection tandis que la deuxième ne peut le faire.

Prenons les deux courbes de Bézier définies par les ensembles de points de contrôle $\boldsymbol{R}^1 = \{(0,0),(-4,4),(-1,8)\}$ et $\boldsymbol{R}^0 = \{(0,0),(0,2),(0,4)\}$. Nous pouvons aisément voir qu'elles ne s'intersectent en aucun point autre que $(0,0)$, leur point de contrôle commun (voir la figure 3.5).

Les ensembles $\boldsymbol{Q}'_a$ et $\boldsymbol{Q}'_b$ correspondants sont respectivement

$$\begin{matrix} \boldsymbol{R}_2^1 - \boldsymbol{R}_2^0 & \boldsymbol{R}_1^1 - \boldsymbol{R}_2^0 & \boldsymbol{R}_0^1 - \boldsymbol{R}_2^0 \\ \boldsymbol{R}_1^1 - \boldsymbol{R}_1^0 & \boldsymbol{R}_0^1 - \boldsymbol{R}_1^0 & \\ \cancel{\boldsymbol{R}_0^1 - \boldsymbol{R}_0^0} & & \end{matrix} = \begin{matrix} \text{(-1,4)} & \text{(-4,0)} & \text{(0,-4)} \\ \text{(-4,2)} & \text{(0,-2)} & \end{matrix}$$

et

$$\begin{matrix} \boldsymbol{R}_2^1 - \boldsymbol{R}_2^0 & \boldsymbol{R}_2^1 - \boldsymbol{R}_1^0 & \boldsymbol{R}_2^1 - \boldsymbol{R}_0^0 \\ \boldsymbol{R}_1^1 - \boldsymbol{R}_1^0 & \boldsymbol{R}_1^1 - \boldsymbol{R}_0^0 & \\ \cancel{\boldsymbol{R}_0^1 - \boldsymbol{R}_0^0} & & \end{matrix} = \begin{matrix} \text{(-1,4)} & \text{(-1,6)} & \text{(-1,8)} \\ \text{(-4,2)} & \text{(-4,4).} & \end{matrix}$$

Ces deux ensembles de vecteurs peuvent être séparés de l'origine en utilisant, par exemple, $\boldsymbol{n}_a = (\frac{-1}{10}, 1)$ et $\boldsymbol{n}_b = (\frac{1}{2}, \frac{1}{3})$ comme hyperplans de séparation, des droites dans ce cas-ci (voir la figure 3.6). Nous voyons que le test de la méthode principale est donc concluant.

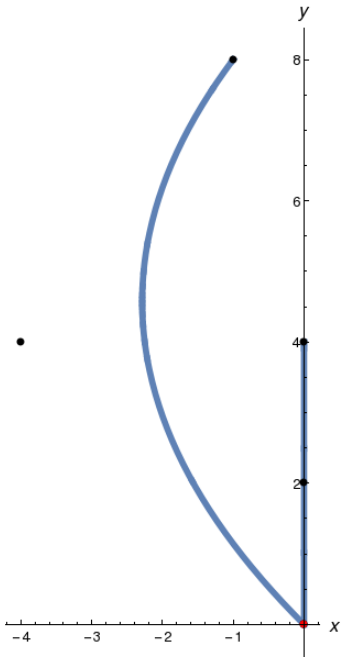

Figure 3.5 – Courbes de Bézier et leurs points de contrôle associés générés par $\boldsymbol{R}^0$ et $\boldsymbol{R}^1$.

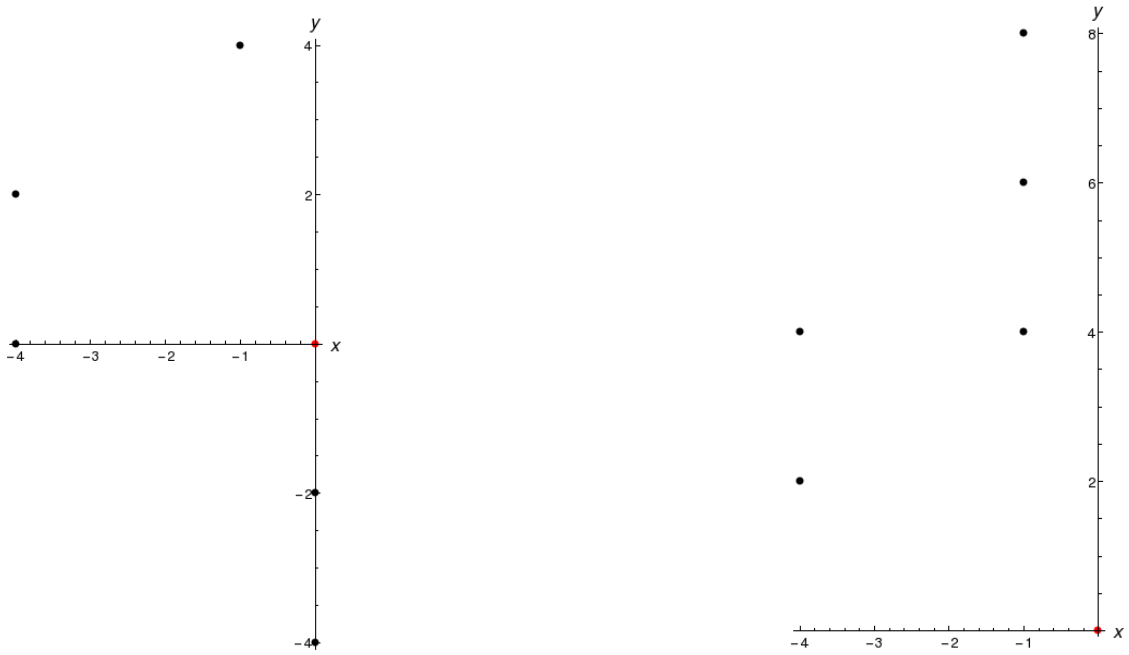

Figure 3.6 – Test-point de l'ensemble $\boldsymbol{Q}'_a$ (gauche) et $\boldsymbol{Q}'_b$ (droite).

Or, les ensembles de test-point de la méthode alternative sont

$$\boldsymbol{F}^a_{12} = -(\boldsymbol{R}^0_1 - \boldsymbol{R}^0_0) \quad -(\boldsymbol{R}^0_2 - \boldsymbol{R}^0_1) = \text{(0,-2)} \quad \text{(0,-2)},$$

$$\boldsymbol{F}^a_{13} = \begin{matrix} (\boldsymbol{R}^1_1 - \boldsymbol{R}^1_0) - (\boldsymbol{R}^0_1 - \boldsymbol{R}^0_0) & (\boldsymbol{R}^1_2 - \boldsymbol{R}^1_1) - (\boldsymbol{R}^0_2 - \boldsymbol{R}^0_1) \\ (\boldsymbol{R}^1_1 - \boldsymbol{R}^1_0) - (\boldsymbol{R}^0_2 - \boldsymbol{R}^0_1) & \end{matrix} = \begin{matrix} \text{(-4,2)} & \text{(3,2)} \\ \text{(-4,2).} & \end{matrix}$$

L'union de $\boldsymbol{F}^a_{12}$ et $\boldsymbol{F}^a_{13}$ ne peut clairement pas être séparée de l'origine (voir la figure 3.7), sans même calculer $\boldsymbol{q}_b$, nous pouvons déjà voir que la méthode alternative ne peut pas réussir à exclure l'intersection.

À partir de ce résultat, il devient clair que la méthode alternative n'a aucun intérêt pratique puisqu'elle est à la fois plus lourde calculatoirement et strictement moins puissante que la méthode principale.

3.2.2 Surfaces

La preuve pour les surfaces, comme mentionné plus tôt, est tout à fait analogue à celle pour les courbes. Cette fois, il faudra par contre distinguer deux cas, deux surfaces ayant une courbe frontière commune et deux surfaces ayant un point de contrôle commun, puisque ce sont les deux cas qui peuvent se présenter dans un complexe bien formé. Débutons par le cas de deux surfaces ayant une courbe frontière commune.

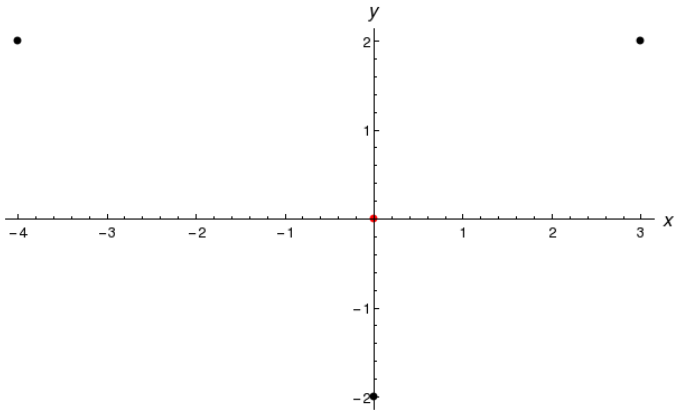

Figure 3.7 – Test-point de l'ensemble $\boldsymbol{q}_a$.

Commençons simplement par remarquer que :

$$\boldsymbol{R}^1_{i+j,m,k+l} = \left\{ \begin{array}{ll} +\boldsymbol{R}^1_{i+j,m,k+l} & -\boldsymbol{R}^1_{i+j-1,m+1,k+l} \\ +\boldsymbol{R}^1_{i+j-1,m+1,k+l} & -\boldsymbol{R}^1_{i+j-2,m+2,k+l} \\ +... & \\ +\boldsymbol{R}^1_{i+1,j+m-1,k+l} & -\boldsymbol{R}^1_{i,j+m,k+l} \end{array} \right\}$$
$$+ \left\{ \begin{array}{ll} +\boldsymbol{R}^1_{i,j+m,k+l} & -\boldsymbol{R}^1_{i,j+m+1,k+l-1} \\ +\boldsymbol{R}^1_{i,j+m+1,k+l-1} & -\boldsymbol{R}^1_{i,j+m+2,k+l-2} \\ +... & \\ +\boldsymbol{R}^1_{i,j+l+m-1,k+1} & -\boldsymbol{R}^1_{i,j+l+m,k} \end{array} \right\} \qquad (3.13)$$
$$+ \left\{ \begin{array}{ll} +\boldsymbol{R}^1_{i,j+l+m,k} & -\boldsymbol{R}^1_{i-1,j+l+m+1,k} \\ +\boldsymbol{R}^1_{i-1,j+l+m+1,k} & -\boldsymbol{R}^1_{i-2,j+l+m+2,k} \\ +... & \\ +\boldsymbol{R}^1_{1,i+j+l+m-1,k} & -\boldsymbol{R}^1_{0,i+j+l+m,k} \\ +\boldsymbol{R}^1_{0,i+j+l+m,k} & \end{array} \right\}$$

et que :

$$-\boldsymbol{R}^0_{i,j+l+m,k} = \left\{ \begin{array}{ll} -\boldsymbol{R}^0_{i,j+l+m,k} & +\boldsymbol{R}^0_{i-1,j+l+m+1,k} \\ -\boldsymbol{R}^0_{i-1,j+l+m+1,k} & +\boldsymbol{R}^0_{i-2,j+l+m+2,k} \\ -... & \\ -\boldsymbol{R}^0_{1,i+j+l+m-1,k} & +\boldsymbol{R}^0_{0,i+j+l+m,k} \\ -\boldsymbol{R}^0_{0,i+j+l+m,k} & \end{array} \right\} \qquad (3.14)$$

En utilisant (3.13), (3.14) et la notation avec l'opérateur de de Casteljau, nous pouvons réécrire tout vecteur $\boldsymbol{R}^1_{i+j,m,k+l} - \boldsymbol{R}^0_{i,j+l+m,k}$ de $\boldsymbol{Q}'_a$ [3, Partie II, (23), p.55 et (10), (12)-

(14), p.48] comme :

$$
\begin{aligned}
&\boldsymbol{R}^1_{i+j,m,k+l} - \boldsymbol{R}^0_{i,j+l+m,k} = \\
&\sum_{J=1}^{n-i-k-l-m} C(1,-1,0)\boldsymbol{R}^1_{i+J-1,n-i-k-l-J,k+l} \\
+&\sum_{L=1}^{n-i-j-k-m} C(0,-1,1)\boldsymbol{R}^1_{i,n-i-k-L,k+L-1} \\
+&\sum_{I=1}^{n-j-k-l-m} C(1,-1,0)\boldsymbol{R}^1_{I-1,n-k-I,k} \\
+&\sum_{I=1}^{n-j-k-l-m} C(-1,1,0)\boldsymbol{R}^0_{I-1,n-k-I,k} \\
&+\boldsymbol{R}^1_{0,i+j+l+m,k} - \boldsymbol{R}^0_{0,i+j+l+m,k}.
\end{aligned}
$$

Les troisième et quatrième sommes peuvent être jointes et les deux derniers éléments peuvent être éliminés puisque leur différence vaut zéro par définition des deux surfaces. Nous avons donc :

$$
\begin{aligned}
&\boldsymbol{R}^1_{i+j,m,k+l} - \boldsymbol{R}^0_{i,j+l+m,k} = \\
&\sum_{J=1}^{n-i-k-l-m} C(1,-1,0)\boldsymbol{R}^1_{i+J-1,n-i-k-l-J,k+l} \\
+&\sum_{L=1}^{n-i-j-k-m} C(0,-1,1)\boldsymbol{R}^1_{i,n-i-k-L,k+L-1} \\
+&\sum_{I=1}^{n-j-k-l-m} \left(C(1,-1,0)\boldsymbol{R}^1_{I-1,n-k-I,k} + C(-1,1,0)\boldsymbol{R}^0_{I-1,n-k-I,k}\right).
\end{aligned}
$$

Remarquons ensuite que les éléments de la première somme sont des éléments de $\boldsymbol{F}^a_{52}$. Pour le voir, il suffit de prendre pour indices de $\boldsymbol{F}^a_{52}$: $i' = i$, $k' = k$, $l' = l$, m' variant entre m et $m + j - 1$ et j' entre $j - 1$ et 0, où les indices primés sont ceux de $\boldsymbol{F}^a_{52}$ et les indices non primés sont ceux des sommes. De la même façon, les éléments de la deuxième somme sont les éléments de $\boldsymbol{F}^a_{54}$ avec les indices : $i' = i,\ j' = 0,\ k' = k,\ m'$ variant entre $m + j$ et $m + j + l - 1$ et l' entre $l - 1$ et 0. Les éléments de la dernière somme correspondent aux éléments de $\boldsymbol{F}^a_{51}$ pour les indices : $j' = 0,\ k' =$

$k,\ l'=0,\ i'$ variant entre $i-1$ et 0 et m' entre $j+l+m$ et $i+j+l+m-1$.

Nous avons donc une représentation des éléments de $\boldsymbol{Q}'_a$ comme une somme de trois sommes qui sont respectivement des sommes d'éléments de $\boldsymbol{F}^a_{52}$, $\boldsymbol{F}^a_{54}$ et $\boldsymbol{F}^a_{51}$. Des dérivations similaires peuvent être faites pour écrire les éléments de $\boldsymbol{Q}'_\nu,\ \nu=b,c,d,e,f$ en termes des éléments des $\boldsymbol{q}_\nu$ correspondants. Puisque les éléments de $\boldsymbol{Q}'_\nu$ peuvent s'écrire comme des sommes d'éléments de $\boldsymbol{q}_\nu$ il est clair que si l'ensemble $\boldsymbol{q}_\nu$ peut être séparé de l'origine, *i.e.*, il existe $\boldsymbol{n}_\nu$ tel que pour tout $\boldsymbol{x}\in\boldsymbol{q}_\nu,\ \boldsymbol{x}\cdot\boldsymbol{n}_\nu>0$, alors l'ensemble $\boldsymbol{Q}'_\nu$ pourra l'être aussi, par distributivité du produit scalaire. Comme ceci est vrai pour tout ν, il s'ensuit que si le test $\boldsymbol{q}_\nu$ est concluant pour assurer la non-intersection, alors le test $\boldsymbol{Q}'_\nu$ l'est aussi.

Passons maintenant à la deuxième possibilité, soit celle de deux surfaces ayant un point de contrôle commun. Tout vecteur de $\boldsymbol{Q}'_a$, $\boldsymbol{R}^1_{i+j,m,k+l}-\boldsymbol{R}^0_{i,j+l+m,k}$ peut s'écrire

comme :

$$
\begin{aligned}
&\boldsymbol{R}^1_{i+j,m,k+l} - \boldsymbol{R}^0_{i,j+l+m,k} = \\
&\left\{\begin{array}{ll} +\boldsymbol{R}^1_{i+j,m,k+l} & -\boldsymbol{R}^1_{i+j+1,m,k+l-1} \\ +\boldsymbol{R}^1_{i+j+1,m,k+l-1} & -\boldsymbol{R}^1_{i+j+2,m,k+l-2} \\ +... & \\ +\boldsymbol{R}^1_{i+j+k-1,m,l+1} & -\boldsymbol{R}^1_{i+j+k,m,l} \end{array}\right\} + \left\{\begin{array}{ll} -\boldsymbol{R}^0_{i,j+l+m,k} & +\boldsymbol{R}^0_{i+1,j+l+m,k-1} \\ -\boldsymbol{R}^0_{i+1,j+l+m,k-1} & +\boldsymbol{R}^0_{i+2,j+l+m,k-2} \\ -... & \\ -\boldsymbol{R}^0_{i+k-1,j+l+m,1} & +\boldsymbol{R}^0_{i+k,j+l+m,0} \end{array}\right\} \\
&+\left\{\begin{array}{ll} +\boldsymbol{R}^1_{i+j+k,m,l} & -\boldsymbol{R}^1_{i+j+k+1,m-1,l} \\ +\boldsymbol{R}^1_{i+j+k+1,m-1,l} & -\boldsymbol{R}^1_{i+j+k+2,m-2,l} \\ +... & \\ +\boldsymbol{R}^1_{i+j+k+m-1,1,l} & -\boldsymbol{R}^1_{i+j+k+m,0,l} \end{array}\right\} + \left\{\begin{array}{ll} -\boldsymbol{R}^0_{i+k,j+l+m,0} & +\boldsymbol{R}^0_{i+k+1,j+l+m-1,0} \\ -\boldsymbol{R}^0_{i+k+1,j+l+m-1,0} & +\boldsymbol{R}^0_{i+k+2,j+l+m-2,0} \\ -... & \\ -\boldsymbol{R}^0_{i+k+m-1,j+l+1,0} & +\boldsymbol{R}^0_{i+k+m,j+l,0} \end{array}\right\} \\
&+\left\{\begin{array}{ll} +\boldsymbol{R}^1_{i+j+k+m,0,l} & -\boldsymbol{R}^1_{i+j+k+m+1,0,l-1} \\ +\boldsymbol{R}^1_{i+j+k+m+1,0,l-1} & -\boldsymbol{R}^1_{i+j+k+m+2,0,l-2} \\ +... & \\ +\boldsymbol{R}^1_{i+j+k+m+l-1,0,1} & -\boldsymbol{R}^1_{i+j+k+l+m,0,0} \end{array}\right\} + \left\{\begin{array}{ll} -\boldsymbol{R}^0_{i+k+m,j+l,0} & +\boldsymbol{R}^0_{i+k+m+1,j+l-1,0} \\ -\boldsymbol{R}^0_{i+k+m+1,j+l-1,0} & +\boldsymbol{R}^0_{i+k+m+2,j+l-2,0} \\ -... & \\ -\boldsymbol{R}^0_{i+k+l+m-1,j+1,0} & +\boldsymbol{R}^0_{i+k+l+m,j,0} \end{array}\right\} \\
&+\boldsymbol{R}^1_{i+j+k+l+m,0,0} \\
&+\left\{\begin{array}{ll} -\boldsymbol{R}^0_{i+k+l+m,j,0} & +\boldsymbol{R}^0_{i+k+l+m+1,j-1,0} \\ -\boldsymbol{R}^0_{i+k+l+m+1,j-1,0} & +\boldsymbol{R}^0_{i+k+l+m+2,j-2,0} \\ -... & \\ -\boldsymbol{R}^0_{i+j+k+l+m-1,1,0} & +\boldsymbol{R}^0_{i+j+k+l+m,0,0} \end{array}\right\} \\
&-\boldsymbol{R}^0_{i+j+k+l+m,0,0}.
\end{aligned}
$$

Comme précédemment, cette dernière expression peut se réécrire en termes de

sommes utilisant l'opérateur de de Casteljau,

$$
\begin{aligned}
&\boldsymbol{R}^1_{i+j,m,k+l} - \boldsymbol{R}^0_{i,j+l+m,k} = \\
&\sum_{K=1}^{n-i-j-l-m} C(-1,0,1)\boldsymbol{R}^1_{n-l-m-K,m,l+K-1} + C(1,0,-1)\boldsymbol{R}^0_{n-j-l-m-K,j+l+m,K-1} \\
&+ \sum_{M=1}^{n-i-j-k-l} C(-1,1,0)\boldsymbol{R}^1_{n-l-M,M-1,l} + C(1,-1,0)\boldsymbol{R}^0_{n-j-l-M,j+l+M-1,0} \\
&+ \sum_{L=1}^{n-i-j-k-m} C(-1,0,1)\boldsymbol{R}^1_{n-L,0,L-1} + C(1,-1,0)\boldsymbol{R}^0_{n-j-L,j+L-1,0} \\
&+ \sum_{J=1}^{n-i-k-l-m} C(1,-1,0)\boldsymbol{R}^0_{n-J,J-1,0} \\
&+ \boldsymbol{R}^1_{i+j+k+l+m,0,0} - \boldsymbol{R}^0_{i+j+k+l+m,0,0}.
\end{aligned}
$$

Comme dans le cas de surfaces avec courbe commune, on peut voir que les éléments des quatre sommes sont respectivement des éléments de $\boldsymbol{F}^a_{13}$, $\boldsymbol{F}^a_{15}$, $\boldsymbol{F}^a_{14}$ et $\boldsymbol{F}^a_{12}$ et les deux derniers vecteurs somment à zéro. La première somme correspond aux éléments de $\boldsymbol{F}^a_{13}$ avec les indices : i' variant entre $i+k-1$ et i, $j'=j$, k' variant entre 0 et $k-1$, $l'=l$ et $m'=m$. La deuxième somme correspond aux éléments de $\boldsymbol{F}^a_{15}$ d'indices : i' variant entre $i+k+m-1$ et $i+k$, $j'=j$, $k'=0$, $l'=l$, et m' variant entre 0 et $m-1$. La troisième somme correspond aux éléments de $\boldsymbol{F}^a_{14}$ d'indices : i' variant entre $i+k+l+m-1$ et $i+k+m$, $j'=j$, $k'=0$, l' variant entre 0 et $l-1$ et $m'=0$. Finalement, la dernière somme correspond aux éléments de $\boldsymbol{F}^a_{12}$ d'indices : i' variant entre $i+j+k+l+m-1$ et $i+k+l+m$, j' variant entre 0 et $j-1$, $k'=0$, $l'=0$ et $m'=0$.

Nous avons donc le même résultat que précédemment, soit que les vecteurs de $\boldsymbol{Q}'_a$ sont des sommes de vecteurs de $\boldsymbol{q}_a$. Encore une fois, des dérivations similaires peuvent être faites pour écrire les éléments de $\boldsymbol{Q}'_\nu,\ \nu=b,c,d,e,f$ en termes des éléments des $\boldsymbol{q}_\nu$ correspondants.

Les résultats obtenus pour les surfaces avec courbe et point communs, à eux deux, impliquent que lorsque la méthode alternative est concluante, alors la méthode principale l'est aussi, c'est-à-dire qu'elle n'est jamais plus faible.

À nouveau, comme pour les courbes, nous pouvons montrer que la méthode principale n'est pas seulement jamais plus faible, mais aussi parfois plus forte. En prenant des extrusions en z des deux courbes de la section précédente, nous obtenons pour réponse avec la méthode principale qu'elles sont séparables et avec la méthode alternative qu'elles pourraient être inséparables.

Pour les surfaces ayant une courbe frontière commune, nous pouvons prendre par exemple (voir la figure 3.8) :

$$\begin{array}{ccccc} (0,0,0) & (0,0,2) & (0,0,4) & & (0,0,0) & (0,0,2) & (0,0,4) \\ \boldsymbol{R}^0_{002} & \boldsymbol{R}^0_{101} & \boldsymbol{R}^0_{200} & & \boldsymbol{R}^1_{002} & \boldsymbol{R}^1_{101} & \boldsymbol{R}^1_{200} \\ (-4,4,0) & (-4,4,2) & & \text{et} & (0,2,0) & (0,2,2) & \\ \boldsymbol{R}^0_{011} & \boldsymbol{R}^0_{110} & & & \boldsymbol{R}^1_{011} & \boldsymbol{R}^1_{110} & \\ (-1,8,0) & & & & (0,4,0) & & \\ \boldsymbol{R}^0_{020} & & & & \boldsymbol{R}^1_{020} & & \end{array} \tag{3.15}$$

et pour le point de contrôle commun (voir la figure 3.8) :

$$\begin{array}{ccccc} (-1,8,0) & (-1,8,2) & (-1,8,4) & & (0,4,0) & (0,4,2) & (0,4,4) \\ \boldsymbol{R}^0_{002} & \boldsymbol{R}^0_{101} & \boldsymbol{R}^0_{200} & & \boldsymbol{R}^1_{002} & \boldsymbol{R}^1_{101} & \boldsymbol{R}^1_{200} \\ (-4,4,0) & (-4,4,2) & & \text{et} & (0,2,0) & (0,2,2) & \\ \boldsymbol{R}^0_{011} & \boldsymbol{R}^0_{110} & & & \boldsymbol{R}^1_{011} & \boldsymbol{R}^1_{110} & \\ (0,0,0) & & & & (0,0,0). & & \\ \boldsymbol{R}^0_{020} & & & & \boldsymbol{R}^1_{020} & & \end{array} \tag{3.16}$$

Dans les deux cas, pour la méthode principale, il est facile de vérifier que l'origine n'est pas incluse dans les ensembles de test-point obtenus. Or, pour la méthode alternative, nous savons déjà que le test échoue. En effet, les ensembles de test-point incluent les mêmes points que nous avions déjà pour les courbes, puisque ce sont des extrusions. Nous obtenons donc bel et bien que la méthode principale est strictement plus forte que la méthode alternative puisqu'elle n'est jamais plus faible et est parfois plus forte.

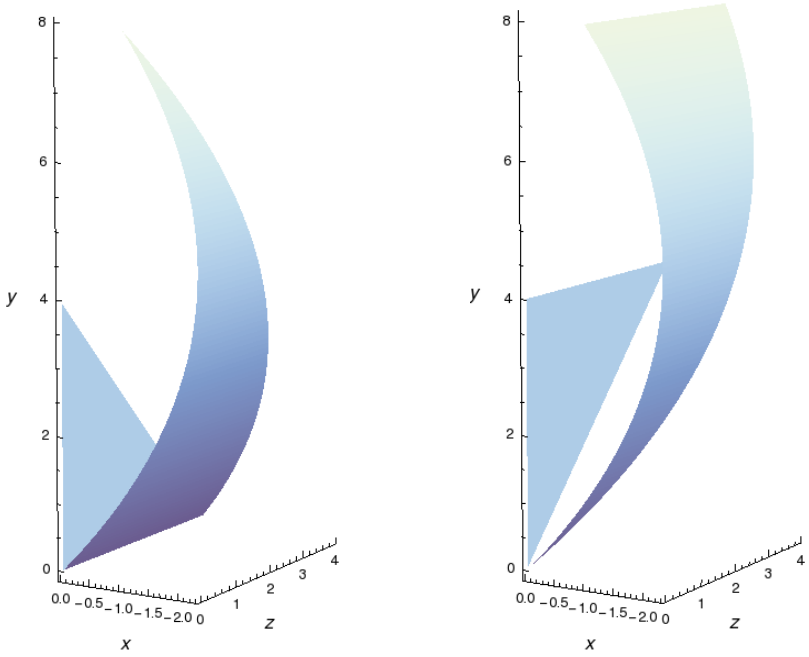

Figure 3.8 – À gauche : les surfaces de Bézier générées par les ensembles de points de contrôle de (3.15) ; à droite : les surfaces de Bézier générées par les ensembles de points de contrôle de (3.16).

3.3 Comparaison entre la méthode de Volino-Thalmann et la méthode test-point pour l'auto-intersection

Nous allons montrer qu'aucune des deux méthodes présentées dans [2] et [33] n'est strictement plus puissante que l'autre. Pour ce faire, il suffit de trouver une surface pour laquelle une méthode est concluante, mais pas l'autre, et vice versa.

3.3.1 La méthode de Volino-Thalmann est parfois plus puissante

Pour montrer que la méthode de [33] est parfois plus puissante que celle de [2], nous allons fournir un exemple de surface où la première réussit, mais la seconde échoue à éliminer la possibilité d'une auto-intersection. Notons toutefois, que pour ce faire, nous montrerons simplement que la méthode Volino-Thalmann *peut* réussir. C'est-à-dire que nous montrerons que les deux conditions principales et les conditions préalables de cette méthode sont effectivement satisfaites. Or, la façon que nous prendrons pour montrer qu'elles peuvent être satisfaites ne serait pas nécessairement celle utilisée par un algorithme purement automatique. Il s'avère que certaines de ces conditions sont difficiles à vérifier de manière automatique. Nous reviendrons sur les particularités et difficultés techniques de la méthode Volino-Thalmann à la section 4.2.

Prenons la surface planaire définie par les points de contrôle suivants :

$$\begin{array}{lll} \underset{\boldsymbol{R}_{002}}{(0,0,0)} & \underset{\boldsymbol{R}_{101}}{(10,0,0)} & \underset{\boldsymbol{R}_{200}}{(20,0,0)} \\ \underset{\boldsymbol{R}_{011}}{(5,40,0)} & \underset{\boldsymbol{R}_{110}}{(15,10,0)} & \\ \underset{\boldsymbol{R}_{020}}{(60,5,0).} & & \end{array} \tag{3.17}$$

Cette surface de Bézier (figure 3.9) est entièrement dans le plan et sa normale ne peut donc avoir une composante non nulle qu'en z. Or, ceci n'est pas suffisant pour garantir la satisfaction de la première condition de la méthode de Volino-Thalmann (voir la section 2.3) puisque le signe de la normale pourrait s'inverser, passant ainsi par zéro. Le test de Volino-Thalmann ne peut donc pas être appliqué tel quel. Il faut d'abord s'assurer

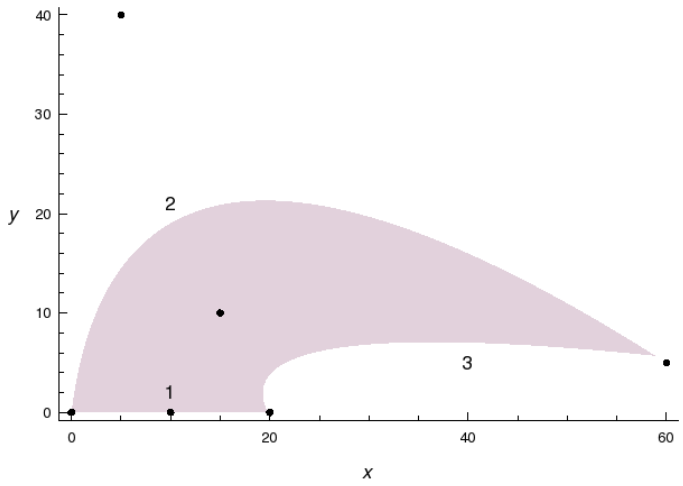

Figure 3.9 – Surface de Bézier générée par les points de contrôle (3.17). Les numéros des courbes font référence à ceux utilisés dans (3.18).

explicitement que la normale ne s'annule en aucun point. L'expression paramétrique de la normale, obtenue en utilisant [5, p.6], est $n_z = 6000r^2 - 2400r - 1200s + 1600$, où r et s sont les coordonnées en espace paramétrique, comme précédemment. Le signe de cette fonction est constant sur tout l'intervalle considéré, soit $r \in [0, 1]$ et $s \in [0, 1 - r]$. La première condition de [33] est donc respectée, il suffit de prendre $\boldsymbol{v} = (0, 0, 1)$.

Pour la deuxième condition, la projection du contour dans le plan perpendiculaire est simplement le contour de la surface puisqu'elle est entièrement dans le plan x-y. Or, la méthode présentée dans l'article original pour s'assurer que la projection du contour ne s'intersecte pas, ne peut être utilisée puisqu'elle n'est applicable que pour des courbes polygonales. Dans l'exemple actuel, étant donné qu'il s'agit de courbes de Bézier, nous pourrions utiliser la méthode test-point pour les courbes présentée dans [2], par exemple. Pour ce faire, il s'agirait de vérifier que chaque courbe ne s'auto-intersecte pas, utilisant [2, Criterion 2.1*], puis que les courbes ne s'intersectent pas entre elles en utilisant [2, Criterion 2.2*]. Or, rappelons encore une fois que nous construisons ici un exemple dans le but explicite de montrer que la méthode Volino-Thalmann peut être plus forte. La question de déterminer comment projeter les courbes et comment vérifier qu'elles ne s'intersectent pas n'est pas toujours évidente et constituerait une difficulté certaine de l'implémentation.

Les trois courbes frontières de la surface définie par (3.17) sont des courbes de Bézier ayant pour points de contrôle les trois ensembles suivants :

$$\begin{aligned} \boldsymbol{Q}_1 &= \{(0,0),(10,0),(20,0)\}, \\ \boldsymbol{Q}_2 &= \{(0,0),(5,40),(60,5)\}, \\ \boldsymbol{Q}_3 &= \{(60,5),(15,10),(20,0)\}. \end{aligned} \tag{3.18}$$

D'abord, pour les tests d'auto-intersection, nous trouvons les trois ensembles de test-point suivants :

$$\begin{aligned} \boldsymbol{q}_1 &= \{(10,0)\}, \\ \boldsymbol{q}_2 &= \{(55,-35),(5,40)\}, \\ \boldsymbol{q}_3 &= \{(5,-10),(-45,5)\}, \end{aligned}$$

qui sont tous trois séparables de l'origine, assurant ainsi la non-auto-intersection. Ensuite pour les intersections deux à deux, nous obtenons :

$$\begin{aligned} \boldsymbol{P}_{12a} &= \{(-5,40),(40,5),(-10,0),(-15,40),(-20,0)\}, \\ \boldsymbol{P}_{12b} &= \{(-5,40),(40,5),(5,40),(50,5),(60,5)\}, \\ \boldsymbol{P}_{13a} &= \{(5,10),(60,5),(10,0),(15,10),(20,0)\}, \\ \boldsymbol{P}_{13b} &= \{(5,10),(60,5),(-5,10),(50,5),(40,5)\}, \\ \boldsymbol{P}_{23a} &= \{(10,-30),(20,0),(55,-35),(15,10),(60,5)\}, \\ \boldsymbol{P}_{23b} &= \{(10,-30),(20,0),(-45,5),(15,-40),(-40,-5)\}, \end{aligned}$$

où les indices réfèrent aux courbes étudiées. Par exemple, les ensembles $\boldsymbol{P}_{12a}$ et $\boldsymbol{P}_{12b}$ sont les deux ensembles de test-point testant la non-intersection entre les courbes de $\boldsymbol{Q}_1$ et $\boldsymbol{Q}_2$, etc. De plus, pour avoir des courbes correspondant aux définitions de l'article original [2], l'ordre des points de contrôle a été inversé sur les deux courbes pour $\boldsymbol{P}_{13a}$ et $\boldsymbol{P}_{13b}$ et sur $\boldsymbol{Q}_2$ pour $\boldsymbol{P}_{23a}$ et $\boldsymbol{P}_{23b}$.

Tous ces ensembles de test-point sont séparables de l'origine, sauf $\boldsymbol{P}_{23b}$. Pour vérifier la non-intersection entre les courbes de $\boldsymbol{Q}_2$ et $\boldsymbol{Q}_3$, il faudra donc utiliser une autre méthode. Dans le cas actuel, il serait possible soit de subdiviser les deux courbes et

de relancer les tests sur les segments deux à deux, ou alors d'utiliser une méthode plus puissante, la méthode d'implicitation de Sederberg et Parry par exemple [31]. La subdivision a l'avantage qu'elle peut être plus rapide si l'appel récursif s'achève rapidement. Par contre, si ce n'est pas le cas, alors à l'inverse, une méthode plus lourde, mais qui offre un résultat assuré, serait plus efficace. Dans le cadre d'un algorithme de détection purement automatique, la deuxième option serait préférable, puisqu'il est alors assuré que la méthode terminera. Ce n'est pas le cas avec un appel récursif, celui-ci subdiviserait à l'infini dans le cas où il y aurait effectivement une intersection.

Nous pouvons voir qu'il est possible d'exclure la possibilité d'auto-intersection sur la surface générée par les points (3.17) avec la méthode de Volino-Thalmann, bien qu'il ne soit pas nécessairement clair comment le faire de manière automatique. Il n'est donc pas garanti que n'importe quelle implémentation de la méthode réussirait, mais nous voyons qu'il est possible qu'elle puisse réussir. Or, en utilisant la méthode test-point, nous ne pouvons pas conclure qu'il n'y a pas d'auto-intersection, puisque nous obtenons les trois ensembles suivants :

$$\begin{aligned}
\boldsymbol{q}_a &= \{(-10,0),(-5,40),(-10,30),(45,-5),(-5,10)\},\\
\boldsymbol{q}_b &= \{(-45,5),(-55,35),(-5,-10),(5,-10),(5,-40),(-5,40)\},\\
\boldsymbol{q}_c &= \{(5,40),(5,10),(55,-35),(10,-30),(10,0)\}.
\end{aligned} \tag{3.19}$$

L'origine est incluse dans l'enveloppe convexe de $\boldsymbol{q}_a$, le test est donc non concluant. Nous avons donc un exemple montrant que le test de [33] peut parfois être plus puissant que celui de [2].

3.3.2 La méthode test-point est parfois plus puissante

Pour montrer que la méthode de [2] est parfois plus puissante que celle de [33], nous allons fournir un exemple de surface où la première réussit, alors que la seconde échoue à éliminer la possibilité d'une auto-intersection. Prenons la surface définie par les points de contrôle suivants :

$$
\begin{array}{cccc}
\text{(-7,-7,-10)} & \text{(0,-15,0)} & \text{(15,-15,0)} & \text{(14,-7,-10)} \\
\boldsymbol{R}_{003} & \boldsymbol{R}_{102} & \boldsymbol{R}_{201} & \boldsymbol{R}_{300} \\
\text{(-15,0,0)} & \text{(0,0,0)} & \text{(15,0,0)} & \\
\boldsymbol{R}_{012} & \boldsymbol{R}_{111} & \boldsymbol{R}_{210} & \\
\text{(-15,15,0)} & \text{(0,15,0)} & & \\
\boldsymbol{R}_{021} & \boldsymbol{R}_{120} & & \\
\text{(-7,14,-10).} & & & \\
\boldsymbol{R}_{030} & & &
\end{array}
\tag{3.20}
$$

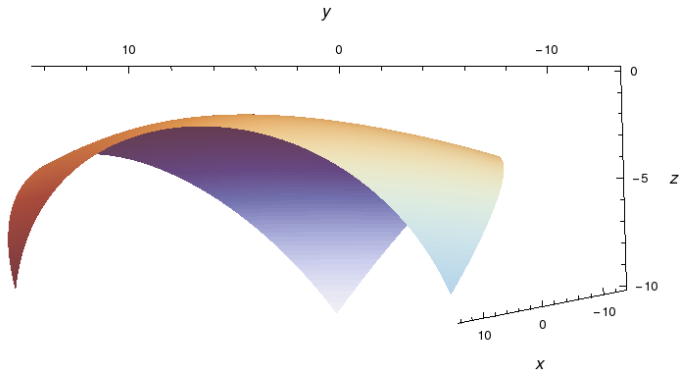

Figure 3.10 – Surface de Bézier générée par les points de contrôle (3.20).

Cette surface ne s'auto-intersecte clairement pas (voir la figure 3.10). De plus, il existe des plans permettant de séparer les enveloppes convexes des trois ensembles de test-point de l'origine (voir les figures 3.11, 3.12 et 3.13). La méthode [2] est donc concluante et elle garantit la non-auto-intersection.

Toutefois, la méthode [33] ne peut pas garantir la non-auto-intersection, même en assumant la plus grande précision atteignable théoriquement. Comme discuté plus haut (section 2.3), ceci serait obtenu en prenant un nombre d'échantillons tendant vers l'infini. Or, même dans ce cas, la méthode échoue. En effet, prenons les vecteurs normaux de la surface évalués à $\boldsymbol{R}(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, $\boldsymbol{R}(0, 0, 1)$, $\boldsymbol{R}(1, 0, 0)$ et $\boldsymbol{R}(0, 1, 0)$. Nous avons les vecteurs

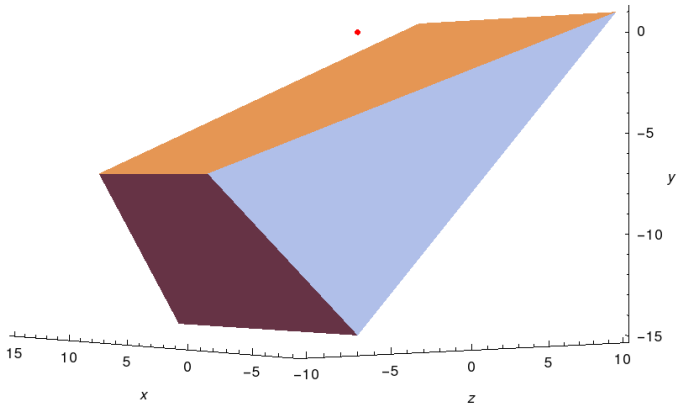

Figure 3.11 – Enveloppe convexe de l'ensemble de test-point $\boldsymbol{q}_a$ de la surface de Bézier générée par les points de contrôle (3.20) avec l'origine en rouge.

suivants :

$$\begin{aligned}
\boldsymbol{n}\left(\frac{1}{3},\frac{1}{3},\frac{1}{3}\right) &\approx (-0.2525,-0.2525,0.9341)\\
\boldsymbol{n}(0,0,1) &\approx (-0.707,-0.707,0.0177)\\
\boldsymbol{n}(1,0,0) &\approx (0.995,0,-0.0995)\\
\boldsymbol{n}(0,1,0) &\approx (0,0.995,-0.0995).
\end{aligned}$$

En suivant la méthodologie présentée en [33, sec. 2.3.4], nous définissons, pour chacun des vecteurs normaux, le demi-espace autour du vecteur dans lequel doit se trouver le vecteur $\boldsymbol{v}$ si nous voulons respecter le premier critère de cette méthode. En prenant l'intersection des intervalles sur toutes les normales, nous obtenons donc l'intervalle où ce vecteur $\boldsymbol{v}$ peut être choisi. Dans l'exemple précédent, l'intersection des quatre intervalles est vide, c'est donc qu'il n'existe aucun vecteur $\boldsymbol{v}$ valide. La méthode [33] ne peut donc pas garantir la non-auto-intersection.

Il existe un exemple équivalent pour des surfaces de degré 2. Nous avons choisi d'utiliser un exemple avec $n = 3$ ici, seulement parce qu'il nous semblait plus clair en prenant la normale aux points de contrôle que l'échec de la méthode ne provient pas de l'évaluation, mais bien d'une limitation intrinsèque de la méthode [33].

Les sections 3.3.1 et 3.3.2 montrent qu'aucune des deux méthodes [2] ou [33] n'est absolument meilleure que l'autre. Il y a des situations où il sera préférable d'utiliser

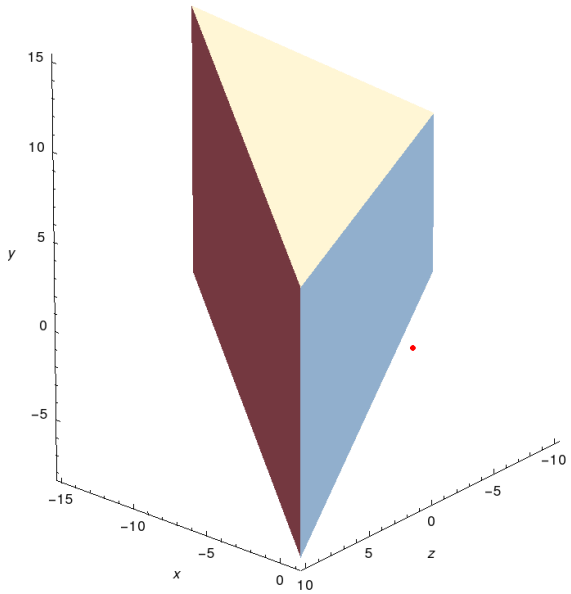

Figure 3.12 – Enveloppe convexe de l'ensemble de test-point $\boldsymbol{q}_b$ de la surface de Bézier générée par les points de contrôle (3.20) avec l'origine en rouge.

l'une ou l'autre, mais ce choix n'est pas évident *a priori*. Nous verrons au chapitre 4 que les deux méthodes ont aussi des différences de coût qu'il faut considérer.

3.4 Utilisation de la subdivision

La subdivision, comme précédemment mentionné, est utilisée dans les algorithmes de haut niveau pour simplifier et raffiner les tests. Elle permet un gain de puissance pour toutes les méthodes vues jusqu'à maintenant. En effet, l'utilisation d'une méthode sur les sous-surfaces générées après subdivision n'est jamais moins puissante que l'utilisation de cette même méthode sur la surface avant subdivision. Dans notre cas, nous nous intéressons aux surfaces de Bézier et pour s'en convaincre, nous n'avons qu'à nous rappeler que les surfaces de Bézier sont un cas particulier des Box-splines. Les points de contrôle de ceux-ci convergent uniformément vers leur surface limite associée, comme montré dans [4, p.205]. Par contre, rappelons-nous aussi que la subdivision, bien que très utile et bien qu'elle augmente la puissance, a aussi un coût important. En effet à

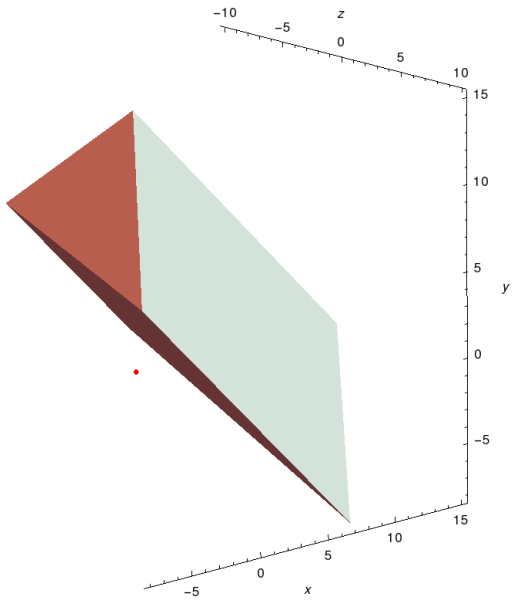

Figure 3.13 – Enveloppe convexe de l'ensemble de test-point $\boldsymbol{q}_c$ de la surface de Bézier générée par les points de contrôle (3.20) avec l'origine en rouge.

chaque étape effectuée, nous augmenterons d'un facteur multiplicatif le nombre de tests à faire, ce qui implique une croissance exponentielle en complexité. Pour le test d'auto-intersection, utilisant l'algorithme de subdivision de Goldman [15] par exemple, nous créerons $C_1^4 + C_2^4 = 10$ nouveaux tests à l'étape actuelle pour chaque sous-surface de l'étape précédente. Ceci découle du fait que chaque nouvelle sous-surface doit être testée pour l'auto-intersection, puis chaque paire de sous-surfaces créées doivent être testées pour l'intersection entre elles.

Regardons en détail les gains en puissance pour les différentes méthodes.

3.4.1 Séparation de volumes englobants

La méthode de la séparation de volumes englobants a avantage à utiliser la subdivision puisque les volumes englobants seront plus petits, ils engloberont donc plus étroitement la surface étudiée. Il sera alors possible de séparer des surfaces qui ne pouvaient pas l'être avant subdivision. Ceci est vrai à la fois pour les volumes plus simples comme les AABB que pour les volumes plus complexes, comme l'enveloppe convexe.

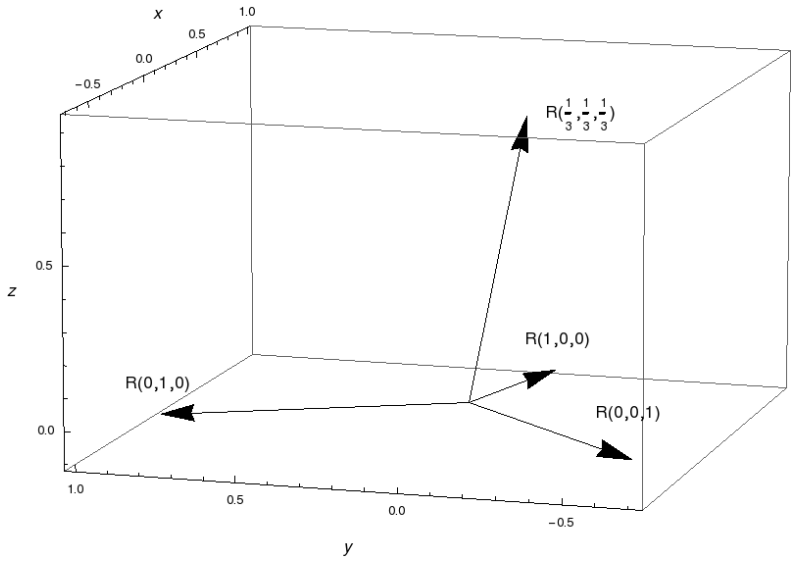

Figure 3.14 – Normales de la surface de Bézier des points (3.20) évaluées à $\boldsymbol{R}\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$, $\boldsymbol{R}(0,0,1)$, $\boldsymbol{R}(1,0,0)$ et $\boldsymbol{R}(0,1,0)$.

Pour la séparation des enveloppes convexes, pour deux courbes, par exemple, reprenons celles de la figure 3.1. Avant subdivision, il est impossible de séparer les deux enveloppes convexes puisque le point (0,0) est commun aux deux enveloppes. Après subdivision, les enveloppes obtenues sont celles présentées à la figure 3.15. Il n'y a plus de point commun entre les enveloppes et il est maintenant possible de déterminer que les deux courbes sont séparables par la méthode de séparation des enveloppes convexes. Il est possible de construire un exemple analogue pour démontrer que le même résultat est aussi valide pour les surfaces. Il suffit de prendre des extrusions des deux courbes données. Nous avons choisi de montrer l'exemple pour les courbes ici puisqu'il est plus facile à visualiser.

Finalement, notons que le test ne peut pas être moins puissant puisque les points de contrôle des surfaces subdivisées sont nécessairement situées dans l'enveloppe convexe des points originaux. L'union des enveloppes convexes des nouveaux points est donc incluse dans l'enveloppe convexe originale.

Il est possible de faire une preuve tout à fait analogue pour la méthode AABB et les mêmes courbes peuvent être utilisées pour le contre-exemple.

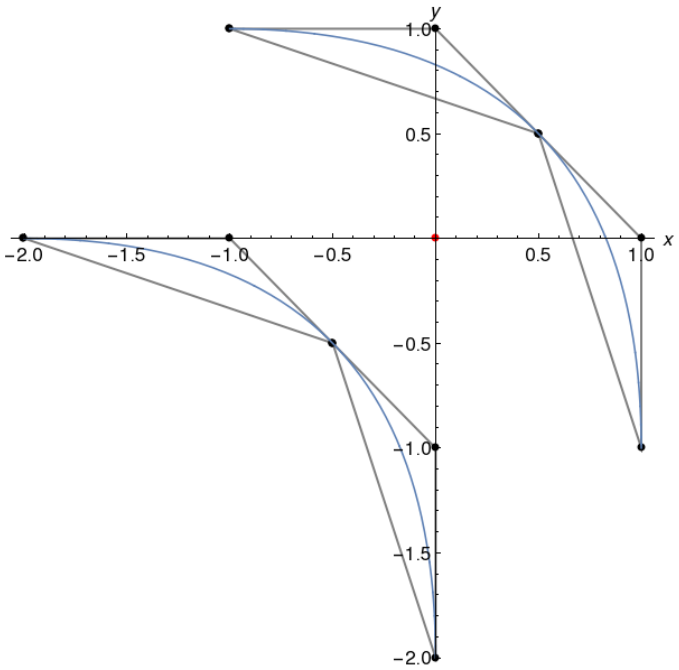

Figure 3.15 – Enveloppes convexes des points de contrôle des deux courbes de Bézier générées par les points de contrôle (3.5) et (3.6) après une étape de subdivision, en subdivisant selon $r = 0.5$.

3.4.2 Méthode test-point

Dans le cas de la méthode test-point, nous n'avons pas une preuve de monotonicité de l'augmentation de la puissance comme pour la méthode précédente. Il n'est pas clair s'il serait possible d'obtenir une telle preuve, puisque après chaque nouvelle étape de subdivision, nous créons de nouveaux tests d'auto-intersection, mais aussi de nouveaux tests pour l'intersection de sous-surfaces adjacentes et distinctes. Ces tests n'opérant pas sur les mêmes différences de points de contrôle, il pourrait être possible que le test après subdivision ne soit pas strictement plus puissant qu'avant. Or, nous savons tout de même que, à la limite, les points de contrôle convergent sur la surface, par le théorème de convergence uniforme que nous avons évoqué plus tôt. Les sous-surfaces deviendront donc de plus en plus planaires à mesure que l'on subdivise. Ceci garantit que le processus de subdivision augmentera éventuellement la puissance.

Comme nous le verrons avec les résultats d'implémentation dans la prochaine

section, la subdivision augmente effectivement de beaucoup la puissance. Prenons, par exemple, la courbe de Bézier générée par les points suivants pour illustrer cette augmentation de la puissance :

$$\underset{\boldsymbol{R}_0}{(-1,2)} \quad \underset{\boldsymbol{R}_1}{(-4,0)} \quad \underset{\boldsymbol{R}_2}{(4,0)} \quad \underset{\boldsymbol{R}_3}{(1,2).} \tag{3.21}$$

Cette courbe (figure 3.16) ne s'auto-intersecte pas, mais la méthode test-point échoue à garantir la non-auto-intersection. Or, comme nous pouvons le voir sur la figure 3.17, après subdivision, les enveloppes convexes des deux nouveaux ensembles de test-point pour l'auto-intersection sont séparables de l'origine, ainsi que ceux pour l'intersection des nouvelles courbes entre elles. Le test pour l'intersection des deux nouvelles courbes entre elles est aussi concluant.

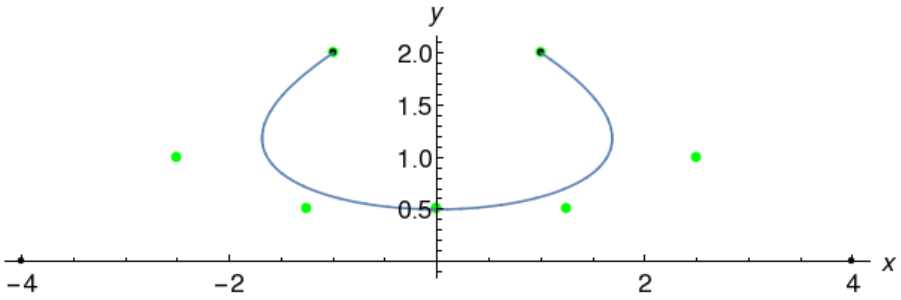

Figure 3.16 – Courbe de Bézier générée par les points de contrôle (3.21). En bleu : la courbe ; en noir : les points de contrôle originaux ; en vert : les points de contrôle obtenus après une étape de subdivision, en subdivisant selon $r = 0.5$.

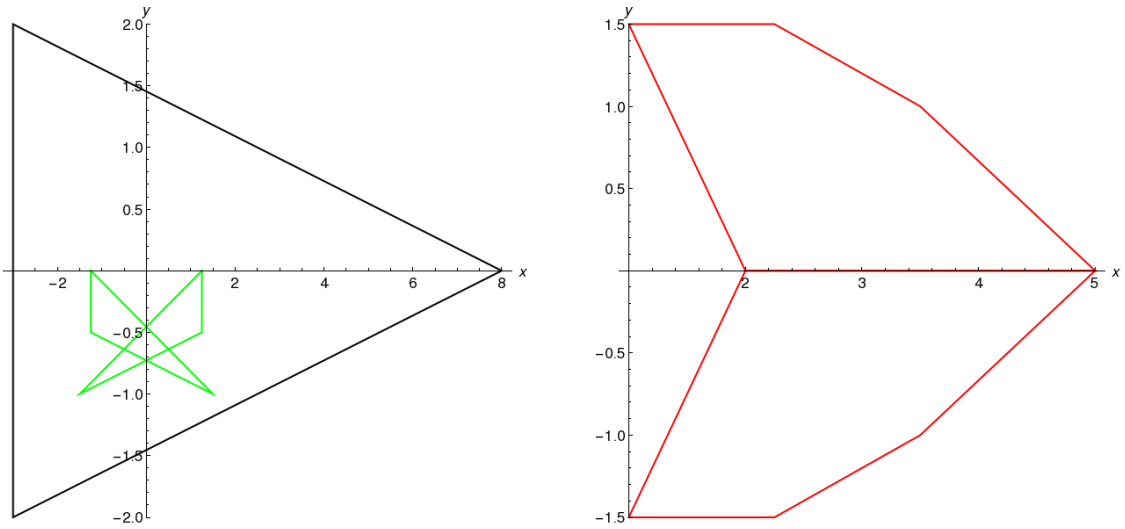

Figure 3.17 – Enveloppes convexes des test-point de la courbe générée par les points de contrôle (3.21). À gauche : en noir : l'enveloppe convexe des test-point de la courbe originale ; en vert : les deux enveloppes convexes des test-point pour l'auto-intersection des courbes obtenues après une étape de subdivision, en subdivisant selon $r = 0.5$. À droite : les enveloppes des test-point vérifiant l'intersection des deux courbes obtenues après subdivision entre elles.

Comme à la section précédente, une extrusion du même exemple peut être utilisée pour montrer que le résultat obtenu tient aussi pour les surfaces. Encore une fois, nous avons choisi de montrer un exemple de courbe puisqu'il nous semblait plus facile à visualiser.

Tests empiriques

En plus de remarquer que la subdivision augmente théoriquement la puissance de la méthode test-point, nous avons voulu savoir quel était le gain de puissance pratique obtenu avec la subdivision. Pour ce faire, nous avons généré un grand nombre de surfaces de Bézier triangulaires cubiques pseudo-aléatoires, desquelles nous avons conservé seulement celles qui ne s'auto-intersectaient pas. Pour ce faire, nous avons subdivisé jusqu'à obtenir des sous-surfaces de taille de l'ordre de 10^{-14} et si nous trouvions toujours une auto-intersection présumée à ce niveau de subdivision, nous considérions qu'il y en avait effectivement une. Parmi les surfaces ainsi générées, nous avons testé

pour l'auto-intersection et, dans le cas où la méthode n'arrivait pas à garantir la non-intersection, nous avons subdivisé avec l'algorithme de Goldman [15] et réappliqué le test et l'étape de subdivision jusqu'à avoir une réponse négative.

Pour construire des surfaces pseudo-aléatoires avec une courbure variable, nous avons généré des points à l'intérieur de boîtes englobantes de tailles variables disposées selon une grille régulière. Ainsi, pour les boîtes de petite taille, les points seront nécessairement plus alignés avec la grille et les surfaces seront plus « plates », tandis que pour les grandes boîtes, les points seront plus éparpillés et la surface résultante sera probablement plus courbée, voir la figure 3.18. Il faut bien noter ici que la surface

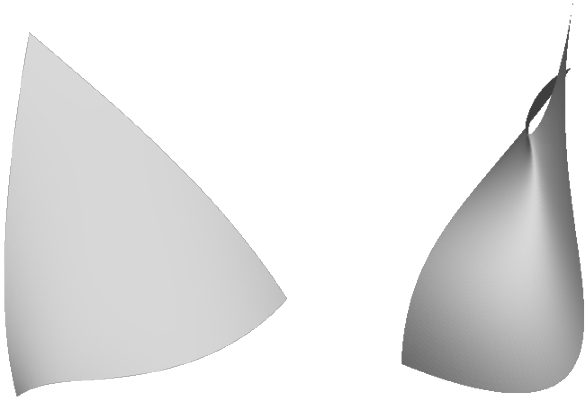

Figure 3.18 – Exemple de surfaces de courbures variées générées en variant le paramètre de taille des boîtes englobantes des points de contrôle. À gauche, le ratio entre la taille et la distance des boîtes est de 1,12 et, à droite, de 3,46.

résultante ne sera que *probablement* plus courbée puisque, même avec des grandes boîtes, il est possible que la surface soit quasi planaire si les points générés sont près du centre des boîtes. La taille des boîtes ne peut donc pas être utilisée comme estimateur de la courbure, bien qu'elles soient corrélées. Pour obtenir un estimé de la courbure réelle de chaque surface générée, nous avons fait un échantillonnage de la courbure gaussienne et de la courbure moyenne de la surface limite. Les courbures gaussienne et moyenne peuvent être calculées en utilisant les déterminants de la première et deuxième forme fondamentale, comme expliqué dans [11], par exemple. Nous avons évalué la courbure à deux millions de positions uniformément distribuées sur la surface et gardé quatre

mesures soit, la courbure gaussienne moyenne, la courbure gaussienne maximale, la courbure moyenne moyenne et la courbure moyenne maximale. Nous avons ensuite testé pour l'auto-intersection en utilisant la méthode [2]. Dans le cas de l'échec de la méthode, nous avons subdivisé la surface en quatre sous-surfaces suivant la méthode de [15] et avons relancé le test. Finalement nous avons répété ces deux dernières étapes jusqu'à obtenir la réponse recherchée, soit qu'il n'y a pas d'intersection. Ce que nous avons pu observer en suivant ce processus sur 20 603 surfaces est que le nombre moyen d'étapes de subdivision nécessaires à la résolution en fonction de la courbure suit empiriquement une loi logarithmique, comme on peut le voir à la figure 3.19. Ceci est vrai pour les quatre

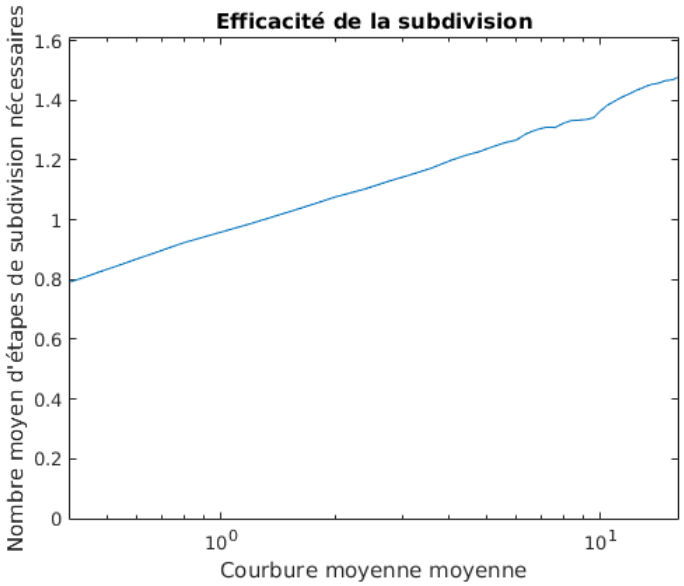

Figure 3.19 – Nombre moyen d'étapes de subdivision nécessaires pour déterminer la non-auto-intersection de surfaces en fonction de leur courbure moyenne moyennée sur 2 millions d'échantillons uniformément distribués.

mesures que nous avons utilisées. Nous voyons donc que la courbure est effectivement un indicateur de la difficulté à éliminer la possibilité d'auto-intersection sur une surface. Or, ce n'est sûrement pas le meilleur indicateur. Il ne permet pas, comme nous l'aurions souhaité, de prédire le nombre d'étapes de subdivision qui sera nécessaire pour arriver à une réponse. De plus, il est passablement coûteux à calculer, puisque la courbure étant une propriété locale, il faut nécessairement l'échantillonner un grand nombre de

fois pour arriver à avoir une idée des propriétés globales de la surface considérée. Nous avons effectivement vu que si nous prenions un petit nombre d'échantillons, nous « manquions » souvent des plis ou des caractéristiques importantes de la surface. Une mesure plus globale du maillage de contrôle ferait probablement mieux l'affaire. Nous pourrions par exemple utiliser une des méthodes présentées dans [23], qui fait une très bonne recension des méthodes existantes pour estimer la courbure gaussienne sur des maillages triangulaires. Nous avons, pour fin de comparaison, implémenté l'estimation de courbure présentée dans [24], soit la version discrète de la méthode de Gauss-Bonnet. Les résultats obtenus étaient très similaires à ceux que nous avions avec notre échantillonnage de la courbure gaussienne et moyenne réelle. Le temps de calcul de cet estimé est pourtant beaucoup plus petit. Il semble donc qu'il s'agirait d'un meilleur choix pratique pour avoir une approximation *a priori* de la difficulté à déterminer la non-auto-intersection.

La deuxième observation que nous avons pu faire est qu'il n'était pas utile de subdiviser plus qu'un certain nombre de fois la surface originale. En effet, pour toutes les surfaces, une réponse concluante a pu être trouvée en huit étapes de subdivision ou moins. Du point de vue de la puissance, ce résultat est très fort, bien qu'informel. Ceci revient à dire que bien que le test de [2] ne soit qu'une condition suffisante, si on le combine à un processus de subdivision et qu'on le répète quelques fois, la probabilité d'assumer à tort qu'il y a intersection alors qu'il n'y en a pas est presque nulle. D'un point de vue du coût, ce résultat est aussi très intéressant, puisque l'application de quelques étapes de la méthode test-point et de la subdivision n'est pas très chère, par opposition à une condition qui serait réellement nécessaire et suffisante. Nous reviendrons sur les questions d'efficacité au prochain chapitre.

3.4.3 Méthode Volino-Thalmann

Encore une fois, pour la méthode Volino-Thalmann, un raisonnement analogue permet de voir que la subdivision ne peut qu'augmenter la puissance. En effet, la subdivision aura l'effet de réduire la taille de l'espace paramétrique, ce qui garantit que la normale pourra être bornée par un cône de taille égale ou moindre. Aussi, les différences

de points de contrôle seront plus petites ou égales, ce qui garantit une puissance au moins équivalente pour le test d'intersection des contours si l'on utilise la méthode test-point pour vérifier cette condition.

De plus, nous obtenons, cette fois encore, qu'il existe des surfaces pour lesquelles le test est inconcluant avant subdivision, mais concluant après, prenant la surface de la figure 3.10, par exemple, et en subdivisant par quatre selon la méthode de [15].

3.5 Conclusion

À la lumière de tous les résultats présentés dans cette section, nous pouvons voir qu'individuellement certaines méthodes sont meilleures que d'autres. D'abord, les méthodes de Volino-Thalmann et test-point sont de puissance comparable ou du moins aucune des deux n'est strictement plus puissante que l'autre. Ensuite, la méthode test-point est strictement plus puissante que la méthode de séparation des enveloppes convexes, qui est elle-même strictement plus puissante que la méthode AABB. Cette dernière affirmation tient du fait que ces deux volumes englobants sont convexes et que l'enveloppe convexe des points de contrôle est nécessairement incluse dans l'AABB. Ceci nous donne un portrait de la situation quant aux puissances individuelles, mais qu'en est-il de la puissance « pratique » ? Les méthodes présentées sont généralement utilisées dans un contexte non individuel, elles sont plutôt couplées avec une structure de hiérarchisation et un processus de subdivision. Ces ajouts permettent d'augmenter la puissance individuelle de toutes les méthodes présentées, mais il reste à voir comment ces gains varient d'une méthode à l'autre. Finalement, il reste aussi à voir comment les méthodes diffèrent dans leur coût d'utilisation. C'est ce que nous tenterons de faire dans le prochain chapitre.

CHAPITRE 4

DISCUSSION SUR L'EFFICACITÉ

Nous avons vu au chapitre précédent qu'il existe des différences théoriques de puissance entre les méthodes. Nous avons aussi vu que les méthodes gagnent toutes à utiliser la subdivision d'une façon ou d'une autre. Ces deux constats ouvrent de nouvelles perspectives et de nombreuses questions sur l'efficacité. À coût calculatoire égal, il va sans dire que la méthode la plus puissante est le meilleur choix. Or, les méthodes vues ne sont pas à coûts égaux. Une analyse asymptotique simple nous révélerait déjà qu'elles diffèrent sur ce point, comme nous le verrons à la section 4.1. Dans ce cas, le choix de la méthode à adopter variera selon notre préférence à accorder plus d'importance à la justesse du résultat ou à la rapidité pour l'obtenir. Dans plusieurs domaines, comme nous l'avons déjà mentionné en introduction, le coût est primordial. Dans ce cas, le choix de AABB est généralement fait, puisqu'il est de loin le plus simple. Similairement, pour les cas où la précision est le facteur déterminant, le choix de la méthode la plus puissante serait fait. Cependant, le choix à faire n'est presque jamais aussi tranché et facile. Dans la quasi totalité des contextes, nous voulons une méthode précise, mais le coût est aussi un facteur déterminant. Généralement, aussi, nous allons subdiviser jusqu'à avoir une réponse « certaine » pour une précision donnée. Dans ce cas, la puissance et le coût sont d'autant plus liés. Une méthode plus puissante et pour laquelle il sera nécessaire de subdiviser moins souvent offre un gain en temps puisque la subdivision elle-même a un coût non négligeable et engendre un nombre de tests supplémentaires croissant exponentiellement selon le nombre d'étapes de subdivision. Effectivement, pour chaque sous-surface que nous subdivisons, nous créerons $C_1^4 + C_2^4$ nouveaux tests, pour les tests d'auto-intersection et d'intersection deux à deux, respectivement, si nous subdivisons en utilisant l'algorithme de Goldman, par exemple. Il faut donc tenter de trouver un agencement de méthodes pour tenter d'optimiser à la fois le coût et la puissance. La question de savoir quelle combinaison de méthodes permet d'optimiser le temps de calcul demanderait de faire une étude empirique approfondie. Cette étude,

pour porter des résultats réellement instructifs et applicables, devrait être exhaustive, ce qui représente une recherche dans un espace très grand. Nous croyons qu'une telle étude pourrait constituer le travail d'un mémoire en soi. Or, bien que nous n'ayons pas entrepris cette étude, nous croyons, avec les résultats présentés au chapitre 3, avoir réussi à établir les fondations nécessaires pour l'entreprendre. D'abord, considérant ce que nous avons vu à la section 3.3 et les difficultés de la méthode Volino-Thalmann, sur lesquelles nous reviendrons à la section 4.2, il nous semble que cette méthode montre un potentiel limité quant au coût. Ensuite, il nous semble aussi que les résultats de la section 3.4.2 montrent que la méthode test-point couplée avec la subdivision soit très puissante, ce qui, nous l'espérons, pourrait aussi signifier un coût compétitif par rapport à des méthodes plus simples comme la séparation des enveloppes convexes ou des AABB. Finalement, il nous semble clair qu'une solution optimale ferait appel à plusieurs méthodes et certainement un processus de subdivision.

4.1 Analyse théorique

Une première étape, si nous voulons nous pencher sur la question du coût, sera de regarder théoriquement par une analyse asymptotique quelles sont les différences entre les méthodes individuelles. Comme nous l'avons vu au chapitre précédent, les techniques de boîtes englobantes et d'enveloppes convexes sont strictement moins puissantes que les techniques de Volino-Thalmann et test-point, qui sont toutes deux de puissance potentiellement comparable. Nous nous attendions donc à trouver un temps de calcul suivant ce même ordre partiel, et c'est effectivement ce que nous pouvons observer. Pour tester l'intersection entre des objets distincts, si n est le nombre de points de contrôle :

- AABB prendrait $\Theta(n)$ temps par objet pour construire la boîte, puis $\Theta(1)$ pour le test, soit $\Theta(n)$ temps total.
- La séparation des enveloppes convexes prendrait $\Theta(n \lg n)$ par objet en moyenne pour construire l'enveloppe[1], puis $\Theta(n)$ pour le test, soit $\Theta(n \lg n)$ temps total en moyenne.

[1] Ce serait de $\Theta(n^2)$ dans le pire cas, si les points sont placés sur une sphère, par exemple.

- La méthode test-point prendrait $\Theta(n^2)$ pour construire les ensembles de test-point, $\Theta(n \lg n)$ en moyenne pour construire les enveloppes convexes, puis $\Theta(n)$ pour le test, soit $\Theta(n^2)$ temps total en moyenne.

Pour tester l'intersection sur un seul objet :

- La méthode test-point prendrait $\Theta(n^2)$ pour construire les ensembles de test-point, $\Theta(n \lg n)$ en moyenne pour construire les enveloppes convexes, puis $\Theta(n)$ pour le test, soit $\Theta(n^2)$ temps total en moyenne.

- La méthode Volino-Thalmann prendrait $\Theta(n^2)$ pour borner la normale dans un cône, utilisant [30, sect. 4] pour un estimé conservateur par exemple, puis $\Theta(n)$ pour tester l'intersection du contour avec lui-même en supposant que nous n'utilisions que la méthode test-point pour vérifier cette condition (nous reviendrons sur la question de vérifier la possibilité d'intersection de la projection du contour dans la sous-section 4.2.2), soit $\Theta(n^2)$ temps total.

4.2 Particularités de la méthode Volino-Thalmann

La méthode Volino-Thalmann, telle qu'initialement présentée dans [33], est attrayante d'un point de vue du coût. Selon leurs résultats, elle performe bien mieux que la méthode hiérarchique « classique » avec laquelle ils se comparent. Or, l'application pour laquelle ils l'utilisent n'est pas critique, au sens où une détection d'intersection manquée pourrait être acceptable. La méthode telle qu'ils la décrivent n'est donc pas failsafe. Ils mentionnent même que le test sur la deuxième condition est souvent omis puisqu'elle n'est que très rarement insatisfaite. La généralisation et la formalisation de la méthode présentée dans [5] la rendent effectivement failsafe, mais au coût de rajouter certaines vérifications préalables, ou nouvelles conditions. Ces conditions ne sont pas triviales à vérifier. Elles ont donc un impact significatif sur l'efficacité de la méthode. Regardons en détail ces conditions et les coûts qu'elles impliquent.

4.2.1 Borner la normale

Comme mentionné dans [18], [27] et [33], pour pouvoir éliminer les tests sur des sous-surfaces adjacentes, il faut être capable de borner la normale dans un cône de demi-angle plus petit que $\frac{\pi}{2}$. Ceci, pour les surfaces paramétriques, demande passablement plus de travail que sur les surfaces polygonales. Effectivement, [18] montre la difficulté de réussir à trouver une borne assez précise dans le cas des surfaces de Loop. Le cas des surfaces de Bézier est tout aussi ardu. En effet, et comme nous l'avons mentionné à la section 4.1, en prenant une méthode comme [30] par exemple, ceci demanderait $\Theta(n^2)$ en temps, où n est le nombre de points de contrôle. Ce temps est passablement plus long que ce qui est nécessaire pour les surfaces polygonales, qui prenait plutôt $\Theta(n)$. Or, ce n'est pas tout : pour avoir une méthode failsafe, il n'est pas suffisant de pouvoir borner la normale dans un cône. Il faut aussi s'assurer explicitement qu'elle ne peut pas être le vecteur nul. La vérification de cette autre condition est nécessaire. La surface de la figure 4.1 [1], par exemple, montre comment une surface entièrement dans le plan n'a pas nécessairement une normale qui se comporte bien. Cette surface est celle définie par les points

$$\begin{array}{ccc} \underset{\boldsymbol{R}_{002}}{(-1,-6)} & \underset{\boldsymbol{R}_{101}}{(-\frac{1}{2},-5)} & \underset{\boldsymbol{R}_{200}}{(0,-4)} \\ \underset{\boldsymbol{R}_{011}}{(\frac{3}{2},-3)} & \underset{\boldsymbol{R}_{110}}{(0,0)} & \\ \underset{\boldsymbol{R}_{020}}{(4,0).} & & \end{array} \tag{4.1}$$

Nous pouvons aisément voir sur la figure 4.1 que la surface se replie sur elle-même dans la région autour du point $\boldsymbol{R}_{200}$ et s'auto-intersecte. Nous voyons dans cet exemple que la normale change de signe, passant ainsi par zéro. Il y a donc nécessité de faire ce test explicite, ce qui est un désavantage de cette méthode par rapport à la méthode test-point. Pour cette dernière, il n'est pas nécessaire de vérifier explicitement si la normale s'annule. Lorsque c'est le cas, alors l'origine est nécessairement incluse dans l'enveloppe des test-point. C'est donc dire que la méthode test-point inclut « implicitement » déjà cette condition : elle est testée par défaut. Pour s'en convaincre, il suffit de se rappeler

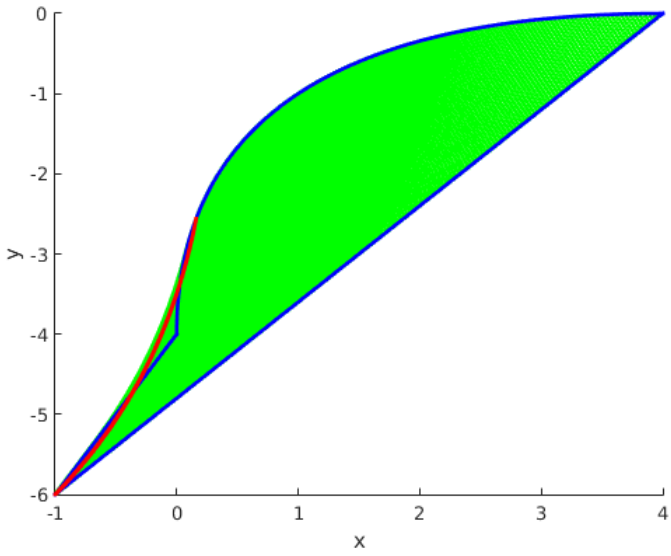

Figure 4.1 – Surface de Bézier planaire où la normale s'inverse. En vert : la surface ; en bleu : les courbes frontières de la surface ; en rouge : la courbe le long de laquelle la normale est nulle.

de la définition de la fonction de laquelle sont issus les test-point que nous avons présentée à la section 2.2. S'il existe un point pour lequel la normale est nulle, alors la dérivée directionnelle de l'application $(r, s, t) \mapsto \boldsymbol{R}(r, s, t)$ sera nulle en ce point. Ceci implique que la fonction construite passera par l'origine. Cette dernière sera donc incluse dans l'enveloppe convexe des test-point et le test donnera le résultat souhaité, soit qu'il y a intersection non désirée. La vérification de cette condition supplémentaire dans la méthode Volino-Thalmann implique un temps de calcul supplémentaire non négligeable. Dans notre implémentation, nous avons vérifié cette condition en utilisant le *Symbolic Math Toolbox* de *MATLAB*. Ce n'est probablement pas la méthode la plus efficace pour y arriver, mais le temps que celle-ci prenait était entre une et deux secondes pour des surfaces de degré deux. Ce temps est de trois à quatre ordres de grandeur plus long que celui nécessaire pour tester l'auto-intersection de la même surface avec notre implémentation de la méthode test-point. C'est un temps qui, pour beaucoup d'applications, serait certainement considéré comme prohibitif.

4.2.2 Test d'intersection de la projection du contour

Comme nous l'avons souligné dans la sous-section 3.3.1, la vérification de la deuxième condition de la méthode Volino-Thalmann peut se faire de plusieurs façons. Chacune de ces façons est plus ou moins précise et plus ou moins coûteuse. L'efficacité finale de la méthode dépendra donc de la méthode choisie pour vérifier la deuxième condition. Dans l'exemple que nous avons montré, nous avons d'abord vérifié avec la méthode test-point, puis avec la méthode d'implicitation de Sederberg dans le cas de l'échec de la méthode test-point. Puisque ces deux méthodes sont failsafe, alors la méthode Volino-Thalmann elle-même demeurerait failsafe. Si nous avions voulu par exemple utiliser une des deux autres méthodes plus simples présentées dans l'article de Sederberg pour tester l'auto-intersection du contour, nous aurions perdu cette propriété. Dans des contextes non critiques, la perte de cette propriété ne serait potentiellement pas un problème, surtout que dans les exemples pratiques, il semble qu'elle ne soit que très rarement non respectée ([18], [27], [33] et [34]). Pour cette raison, elle est souvent omise dans ces contextes. Or, pour ce mémoire, puisque nous nous intéressons seulement à des méthodes failsafe, nous n'étudierons pas ces deux alternatives. Dans ce cas, puisque nous devons effectivement vérifier la deuxième condition, un certain coût lui sera associé. Dans l'exemple que nous avons donné, puisque la méthode test-point n'est qu'une condition suffisante, le coût théorique de répondre à la deuxième condition de Volino-Thalmann serait tout de même celui de la méthode de Sederberg, puisque dans le pire cas nous aurons besoin d'y avoir recours. Cette vérification aurait donc un coût de $\Theta(n^2)$, où n est le nombre de points de contrôle. Évidemment, il serait possible de n'utiliser que la méthode test-point et ainsi ne pas avoir une condition suffisante sur le test du contour, mais même dans ce cas, ce test demande un temps non négligeable. De plus, dans ce cas, l'exemple que nous avons choisi, où la méthode Volino-Thalmann était plus puissante que la méthode test-point, ne serait plus valide, puisque la méthode Volino-Thalmann ne serait alors pas concluante. Il reste à voir s'il est possible de trouver un exemple où la méthode Volino-Thalmann serait plus puissante que la méthode test-point, même sans utiliser une condition nécessaire et suffisante pour la vérification de la

non-auto-intersection du contour.

4.2.3 Surfaces adjacentes

Dans sa version paramétrique failsafe, la méthode Volino-Thalmann a un autre désavantage par rapport à la version originale : la gestion des surfaces adjacentes. Originalement, il ne s'agissait que de joindre les surfaces et d'appliquer les tests pour les deux conditions sur l'union des surfaces, alors que maintenant, il faut aussi vérifier la non-intersection de chacun des contours des sous-surfaces formant l'union. Comme nous l'avons vu à la section précédente, cette condition a un poids calculatoire non négligeable.

4.3 Tests empiriques

Pour ce mémoire, nous avons implémenté une version de la méthode Volino-Thalmann, la méthode test-point, la séparation des enveloppes convexes et la séparation des AABB. Nous avons aussi implémenté le processus de subdivision de Farin et celui de Goldman, ainsi que la structure de haut niveau comportant des appels récursifs aux tests individuels pour tenter de manière adaptative de traiter une surface complexe en travaillant sur ses sous-surfaces. Nous avons fait toutes ces implémentations pour vérifier nos résultats théoriques et pour en apprendre plus sur le gain de puissance possible par la subdivision. Bien que nous ayons eu toutes les implémentations nécessaires pour faire des tests empiriques, nous n'avons pas fait une analyse approfondie sur la question, puisque comme nous l'avons déjà mentionné, cette étude devrait être complète pour avoir une valeur, dans le sens où il faudrait étudier toutes les différentes combinaisons possibles de méthodes. Les questions d'agencer les méthodes dans un algorithme automatique de haut niveau pour optimiser le coût est très complexe, étant donné que l'effet des différents paramètres ne peuvent pas être étudiés indépendamment les uns des autres. En effet, pour qu'un processus automatique puisse déterminer de manière autonome quelle méthode choisir à chaque étape, il faudra d'abord développer des métriques précises et simples à calculer qui soient déterminantes de la propension

de chaque méthode à réussir. Comme nous l'avons vu à la section 3.4.2, la question de trouver de telles métriques n'est pas nécessairement évidente. De plus, même en supposant que nous ayons de telles métriques, les processus de subdivision engendrent un nombre de tests croissant exponentiellement selon le nombre d'étapes, comme nous l'avons déjà mentionné en début de chapitre. Évidemment, dans ce contexte, il ne sera pas possible de tester toutes les possibilités. Une éventuelle étude empirique devra donc commencer par faire un certain nombre de tests ciblés comparant les méthodes deux à deux et ainsi exclure de l'espace de recherche certaines possibilités, sans quoi ce serait certainement trop difficile.

Finalement, en plus de devoir tester toutes les méthodes couplées avec le processus de subdivision, le processus de subdivision lui-même devrait sûrement être étudié pour être optimisé. La subdivision selon un ou des points arbitraires apporte l'avantage qu'il est possible de l'adapter à la surface considérée. Nous croyons donc que les points de subdivision pourraient être choisis de façon à distribuer la courbure plus uniformément dans les sous-surfaces. Ceci, il nous semble, devrait pouvoir permettre d'obtenir une réponse favorable en subdivisant un plus petit nombre de fois, donc plus rapidement. Il reste, par contre, comme pour les questions précédentes, à réussir à trouver des métriques pour arriver à déterminer automatiquement où les points optimaux de subdivision se trouvent.

4.4 Conclusion

Premièrement, l'analyse asymptotique que nous avons faite montre que les différences de puissance que nous avons pu observer au chapitre précédent sont directement reflétées dans le coût. L'ordre partiel reliant les méthodes est en effet le même, soit que les méthodes Volino-Thalmann et test-point sont comparables, mais demandent toutes deux plus de temps que la séparation des enveloppes convexes qui demande elle-même plus de temps que la séparation des AABB.

Deuxièmement, considérant toutes les particularités vues à la section 4.2, il semble que la méthode Volino-Thalmann ne soit potentiellement pas la meilleure pour les

applications critiques puisqu'elle demande beaucoup de travail. L'avantage qu'elle offre dans sa version originale résulte du gain obtenu par l'union des sous-surfaces. Ce gain est plus mitigé dans la version paramétrique. D'un point de vue strictement du coût, il semble qu'elle soit moins performante que la méthode test-point. Ceci, bien que potentiellement étonnant si on se fie aux résultats présentés dans l'article original de Volino et Thalmann, est en fait plutôt sensé. Il faut effectivement se rappeler qu'elle a dû être adaptée pour répondre à de nouvelles exigences. Sa formulation n'était pas paramétrique, et pour le devenir, elle a dû se complexifier. Il est donc assez intuitivement satisfaisant de s'apercevoir qu'une méthode comme la méthode test-point, qui a été initialement conçue dans le but d'être failsafe et de travailler sur des surfaces paramétriques, réussisse mieux dans ce contexte. Rappelons-nous tout de même que toutes ces observations sont limitées. Pour avoir une idée véritablement juste des différences d'efficacité, il faudrait faire une analyse empirique comparative approfondie. Il nous paraît clair qu'une telle étude soit le seul moyen de pouvoir tirer des conclusions utiles sur les questions d'efficacité. Les résultats théoriques ne peuvent avoir pour ce problème qu'une applicabilité mitigée.

CHAPITRE 5

CONCLUSION

Nous avons, avec ce mémoire, tenté de répondre à un certain nombre de questions concernant les méthodes de détection d'intersection sur les surfaces paramétriques. Les réponses que nous avons trouvées, loin de satisfaire notre appétit de compréhension, ne font que l'ouvrir davantage. Il reste beaucoup à faire sur le sujet. Voici donc une liste non exhaustive de directions de recherches futures qui, nous croyons, seraient intéressantes et prometteuses.

5.1 Problèmes ouverts

- Comme nous avons vu au chapitre 3, la subdivision offre des gains en puissance pour toutes les méthodes étudiées. Or, nous n'avons que très peu d'information quantitative sur le gain obtenu par chacune des méthodes. Nous croyons que le gain de puissance lié à subdiviser sera variable d'une méthode à l'autre. Nous croyons aussi que dû à sa plus grande puissance initiale, la méthode test-point gagnera proportionnellement davantage à chaque étape que les méthodes moins puissantes. Si ce résultat s'avère, il entraînerait par le fait même un potentiel gain quant au coût. En effet, bien que la méthode test-point soit plus chère individuellement que la méthode AABB, par exemple, s'il est nécessaire de l'appliquer un plus petit nombre de fois que cette dernière et s'il est nécessaire de subdiviser moins de fois avant d'obtenir un résultat, le choix de cette méthode peut alors devenir intéressant d'un point de vue d'efficacité. Pour s'en convaincre, il suffit de se rappeler que la subdivision elle-même a un coût et que chacune de ses étapes engendre $C_1^n + C_2^n$ tests supplémentaires, pour une subdivision en n sous-surfaces, ce qui croît exponentiellement. La question de quantifier le gain de puissance associé à la subdivision pour chacune des méthodes est donc de grand intérêt. Quantifier ce gain de manière précise serait peut-être trop difficile, mais

il devrait au moins être possible de le borner d'une certaine façon, ce qui serait potentiellement suffisant dans une optique de comparaison.

- Les surfaces de Bézier sont utilisées dans plusieurs applications, mais elles demeurent moins répandues que les NURBS et les *trimmed*-NURBS, qui ont une plus grande flexibilité et sont plus faciles d'utilisation. La question de savoir si la méthode test-point pourrait être généralisée à des NURBS a donc un intérêt pratique. Or, ceci n'est que la première étape. Il faudrait ensuite tenter de généraliser les résultats que nous avons obtenus pour les comparaisons de puissance entre méthodes sur les surfaces de Bézier.

- Au chapitre précédent, nous avons vu les différences dans le temps de calcul asymptotique des différentes méthodes. Cette analyse simple nous permet déjà d'en apprendre un peu sur l'efficacité des méthodes, mais elle n'est pas très précise puisque, dans la plupart des applications, les surfaces traitées sont de bas degré. Dans ce cas, le coût asymptotique peut être trompeur. Pour avoir un portrait plus juste de la situation, il faudrait faire des test pratiques sur les méthodes en faisant l'implémentation la plus efficace possible pour chacune d'elle et tenter de les incorporer dans une structure de haut niveau la plus efficace possible.

- Nous avons comparé la puissance de la méthode test-point pour l'auto-intersection avec la méthode Volino-Thalmann et avons trouvé qu'aucune n'est plus puissante que l'autre, mais qu'en est-il de deux surfaces adjacentes ? Il serait intéressant de voir si le même résultat tient pour la méthode test-point pour deux surfaces partageant une arête et la méthode Volino-Thalmann pour surfaces composées. S'il s'avère qu'une des deux est meilleure, cela pourrait changer les conclusions sur le meilleur choix à faire dans le cadre d'un algorithme de détection global. En effet, comme nous l'avons mentionné précédemment, ces cas sont très fréquemment rencontrés puisque ce sont ceux qui se présentent après subdivision. Ils pourraient donc avoir un poid considérable dans l'efficacité d'un algorithme global.

- Comme nous l'avons suggéré au chapitre précédent, il est possible que la méthode

Volino-Thalmann ne soit jamais plus puissante que la méthode test-point, si nous restreignons le choix de la méthode pour la non-auto-intersection de la projection du contour de la surface. Il suffirait de trouver un exemple où elle est effectivement plus puissante que la méthode test-point, mais sans avoir besoin d'utiliser une méthode nécessaire et suffisante pour la vérification du contour afin de s'assurer que notre résultat tient toujours avec cette contrainte supplémentaire. S'il s'avère qu'un tel exemple ne peut exister, alors cela signifierait que la méthode Volino-Thalmann peut parfois être plus puissante que la méthode test-point, mais coûterait alors probablement plus cher. Ceci reste à vérifier.

- Les surfaces de subdivision de Loop, de par leur adoption marquée dans le milieu de l'animation 3D et leur proximité avec les surfaces de Bézier, seraient de bons candidats pour tenter de généraliser certains des résultats présentés dans ce mémoire. Nous savons que dans le cas régulier, elles se réduisent à des surfaces de Bézier, mais qu'en est-il des cas non réguliers ? Pourrait-on aussi prouver des résultats équivalents pour des surfaces ayant des sommets extraordinaires, c'est-à-dire des sommets avec une valence différente de six dans ce cas ? Pour plus de détails sur les surfaces de subdivision, [4] présente un très bon résumé des définitions et des principaux théorèmes utiles.

BIBLIOGRAPHIE

[1] Lars-Erik Andersson. communications privées, 2014.

[2] Lars-Erik Andersson, Thomas J. Peters et Neil F. Stewart. Selfintersection of composite curves and surfaces. *Computer Aided Geometric Design*, 15(5):507 – 527, 1998. ISSN 0167-8396. URL http://www.sciencedirect.com/science/article/pii/S0167839698000053.

[3] Lars-Erik Andersson, Thomas J. Peters et Neil F. Stewart. Selfintersection of composite curves and surfaces. Rapport technique 1061, Université de Montréal, 1998. URL http://www.sciencedirect.com/science/article/pii/S0167839698000053.

[4] Lars-Erik Andersson et Neil F. Stewart. *Introduction to the Mathematics of Subdivision Surfaces*. SIAM, 2010.

[5] Lars-Erik Andersson, Neil F. Stewart et Malika Zidani. Conditions for use of a non-selfintersection conjecture. *Computer Aided Geometric Design*, 23(7):599–611, octobre 2006. ISSN 0167-8396. URL http://dx.doi.org/10.1016/j.cagd.2006.05.002.

[6] C. Bradford Barber, David P. Dobkin et Hannu Huhdanpaa. The Quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4):469–483, 1996.

[7] Wolfgang Böhm. Generating the Bézier points of B-spline curves and surfaces. *Computer-Aided Design*, 13(6):365–366, 1981.

[8] Jung-Woo Chang, Wenping Wang et Myung-Soo Kim. Efficient collision detection using a dual OBB-sphere bounding volume hierarchy. *Computer Aided Design*, 42 (1):50–57, janvier 2010. ISSN 0010-4485. URL http://dx.doi.org/10.1016/j.cad.2009.04.010.

[9] Bernard Chazelle. An optimal algorithm for intersecting three-dimensional convex polyhedra. *SIAM Journal on Computing*, 21(4):671–696, 1992.

[10] Thomas H. Cormen, Charles Leiserson et Ronald L. Rivest. *Introduction to algorithms*. MIT press, 1990.

[11] Manfredo Perdigao Do Carmo. *Differential geometry of curves and surfaces*, volume 2. Prentice-Hall Englewood Cliffs, 1976.

[12] Gerald Farin. Triangular Bernstein-Bézier patches. *Computer Aided Geometric Design*, 3(2):83–127, 1986.

[13] Gerald Farin. *Curves and surfaces for Computer-Aided Geometric Design : a practical guide*. Academic press, 1988.

[14] Alain Fournier et John Buchanan. Chebyshev polynomials for boxing and intersections of parametric curves and surfaces. Dans *Proceedings of EUROGRAPHICS '94*, pages 127–142, 1994.

[15] Ronald N. Goldman. Subdivision algorithms for Bézier triangles. *Computer-Aided Design*, 15(3):159–166, 1983.

[16] Ronald N. Goldman et Daniel J. Filip. Conversion from Bézier rectangles to Bézier triangles. *Computer-Aided Design*, 19(1):25–27, 1987.

[17] Stefan Gottschalk, Ming C. Lin et Dinesh Manocha. OBBTree : A hierarchical structure for rapid interference detection. Dans *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 171–180, New York, NY, USA, 1996. ACM. ISBN 0-89791-746-4. URL `http://doi.acm.org/10.1145/237170.237244`.

[18] Eitan Grinspun et Peter Schröder. Normal bounds for subdivision-surface interference detection. Dans *Proceedings of the Conference on Visualization '01*, VIS '01,

pages 333–340, Washington, DC, USA, 2001. IEEE Computer Society. ISBN 0-7803-7200-X. URL http://dl.acm.org/citation.cfm?id=601671.601723.

[19] John G. Hocking et Gail S. Young. *Topology*. Reading, Mass., Addison-Wesley, 1961.

[20] Christoph M. Hoffmann. Geometric and Solid Modeling. 1989.

[21] Oluwasanmi O. Koyejo, Nagarajan Natarajan, Pradeep K. Ravikumar et Inderjit S. Dhillon. Consistent binary classification with generalized performance metrics. Dans *Advances in Neural Information Processing Systems*, pages 2744–2752, 2014.

[22] Thomas Larsson. An efficient ellipsoid-OBB intersection test. *Journal of Graphics, GPU, and Game Tools*, 13(1):31–43, 2008.

[23] Evgeni Magid, Octavian Soldea et Ehud Rivlin. A comparison of gaussian and mean curvature estimation methods on triangular meshes of range image data. *Computer Vision and Image Understanding*, 107(3):139–159, 2007.

[24] Mohammed Mostefa Mesmoudi, Leila De Floriani et Paola Magillo. Concentrated curvature for mean curvature estimation in triangulated surfaces. Dans *Computational Topology in Image Context*, pages 79–87. Springer, 2012.

[25] Tomoyuki Nishita, Thomas W. Sederberg et Masanori Kakimoto. Ray tracing trimmed rational surface patches. *SIGGRAPH Comput. Graph.*, 24(4):337–345, septembre 1990. ISSN 0097-8930. URL http://doi.acm.org/10.1145/97880.97916.

[26] Les Piegl et Wayne Tiller. *The NURBS book*. 1997.

[27] Xavier Provot. Collision and self-collision handling in cloth model dedicated to design garments. Dans *Computer Animation and Simulation'97*, pages 177–189. Springer, 1997.

[28] Jean E. Schweitzer. *Analysis and application of subdivision surfaces*. Thèse de doctorat, University of Washington, 1996.

[29] Thomas W. Sederberg. *Implicit and parametric curves and surfaces for Computer Aided Geometric Design*. Thèse de doctorat, 1983.

[30] Thomas W. Sederberg et Ray J. Meyers. Loop detection in surface patch intersections. *Computer Aided Geometric Design*, 5(2):161–171, 1988.

[31] Thomas W. Sederberg et Scott R. Parry. Comparison of three curve intersection algorithms. *Computer-Aided Design*, 18(1):58–63, 1986.

[32] Jos Stam. Evaluation of Loop subdivision surface. *Proceedings CD of SIGGRAPH*, 98, 1998.

[33] Pascal Volino et Nadia Magnenat Thalmann. Efficient self-collision detection on smoothly discretized surface animations using geometrical shape regularity. *Computer Graphics Forum*, 13(3):155–166, 1994. ISSN 1467-8659. URL `http://dx.doi.org/10.1111/1467-8659.1330155`.

[34] Pascal Volino et Nadia Magnenat Thalmann. Collision and self-collision detection : Efficient and robust solutions for highly deformable surfaces. Dans *Computer Animation and Simulation'95*, pages 55–65. Springer, 1995.

Annexe I

Vecteurs de l'ensemble $\boldsymbol{Q}_a$ pour la méthode test-point entre objets distincts avec $n = 3$

Les 35 vecteurs formant l'ensemble de test-point $\boldsymbol{Q}_a$ entre objets distincts pour $n = 3$ est :

$$\boldsymbol{R}^a_{00003} = \boldsymbol{R}^1_{030} - \boldsymbol{R}^0_{030}$$
$$\boldsymbol{R}^a_{00012} = \boldsymbol{R}^1_{021} - \boldsymbol{R}^0_{030}$$
$$\boldsymbol{R}^a_{00021} = \boldsymbol{R}^1_{012} - \boldsymbol{R}^0_{003}$$
$$\boldsymbol{R}^a_{00030} = \boldsymbol{R}^1_{003} - \boldsymbol{R}^0_{003}$$
$$\boldsymbol{R}^a_{00102} = \boldsymbol{R}^1_{021} - \boldsymbol{R}^0_{012}$$
$$\boldsymbol{R}^a_{00111} = \boldsymbol{R}^1_{012} - \boldsymbol{R}^0_{012}$$
$$\boldsymbol{R}^a_{00120} = \boldsymbol{R}^1_{003} - \boldsymbol{R}^0_{012}$$
$$\boldsymbol{R}^a_{00201} = \boldsymbol{R}^1_{012} - \boldsymbol{R}^0_{021}$$
$$\boldsymbol{R}^a_{00210} = \boldsymbol{R}^1_{003} - \boldsymbol{R}^0_{021}$$
$$\boldsymbol{R}^a_{00300} = \boldsymbol{R}^1_{003} - \boldsymbol{R}^0_{030}$$
$$\boldsymbol{R}^a_{01002} = \boldsymbol{R}^1_{120} - \boldsymbol{R}^0_{003}$$
$$\boldsymbol{R}^a_{01011} = \boldsymbol{R}^1_{111} - \boldsymbol{R}^0_{003}$$
$$\boldsymbol{R}^a_{01020} = \boldsymbol{R}^1_{102} - \boldsymbol{R}^0_{003}$$
$$\boldsymbol{R}^a_{01101} = \boldsymbol{R}^1_{111} - \boldsymbol{R}^0_{012}$$
$$\boldsymbol{R}^a_{01110} = \boldsymbol{R}^1_{102} - \boldsymbol{R}^0_{012}$$
$$\boldsymbol{R}^a_{01200} = \boldsymbol{R}^1_{102} - \boldsymbol{R}^0_{021}$$
$$\boldsymbol{R}^a_{02001} = \boldsymbol{R}^1_{210} - \boldsymbol{R}^0_{003}$$
$$\boldsymbol{R}^a_{02010} = \boldsymbol{R}^1_{201} - \boldsymbol{R}^0_{003}$$
$$\boldsymbol{R}^a_{02100} = \boldsymbol{R}^1_{201} - \boldsymbol{R}^0_{012}$$
$$\boldsymbol{R}^a_{03000} = \boldsymbol{R}^1_{300} - \boldsymbol{R}^0_{003}$$
$$\boldsymbol{R}^a_{10002} = \boldsymbol{R}^1_{120} - \boldsymbol{R}^0_{102}$$
$$\boldsymbol{R}^a_{10011} = \boldsymbol{R}^1_{111} - \boldsymbol{R}^0_{102}$$
$$\boldsymbol{R}^a_{10020} = \boldsymbol{R}^1_{102} - \boldsymbol{R}^0_{102}$$
$$\boldsymbol{R}^a_{10101} = \boldsymbol{R}^1_{111} - \boldsymbol{R}^0_{111}$$
$$\boldsymbol{R}^a_{10110} = \boldsymbol{R}^1_{102} - \boldsymbol{R}^0_{111}$$
$$\boldsymbol{R}^a_{10200} = \boldsymbol{R}^1_{102} - \boldsymbol{R}^0_{120}$$
$$\boldsymbol{R}^a_{11001} = \boldsymbol{R}^1_{210} - \boldsymbol{R}^0_{102}$$
$$\boldsymbol{R}^a_{11010} = \boldsymbol{R}^1_{201} - \boldsymbol{R}^0_{102}$$
$$\boldsymbol{R}^a_{11100} = \boldsymbol{R}^1_{201} - \boldsymbol{R}^0_{111}$$
$$\boldsymbol{R}^a_{12000} = \boldsymbol{R}^1_{300} - \boldsymbol{R}^0_{102}$$
$$\boldsymbol{R}^a_{20001} = \boldsymbol{R}^1_{210} - \boldsymbol{R}^0_{201}$$
$$\boldsymbol{R}^a_{20010} = \boldsymbol{R}^1_{201} - \boldsymbol{R}^0_{201}$$
$$\boldsymbol{R}^a_{20100} = \boldsymbol{R}^1_{201} - \boldsymbol{R}^0_{210}$$
$$\boldsymbol{R}^a_{21000} = \boldsymbol{R}^1_{300} - \boldsymbol{R}^0_{201}$$
$$\boldsymbol{R}^a_{30000} = \boldsymbol{R}^1_{300} - \boldsymbol{R}^0_{300}\,.$$

Annexe II

Coefficients ν et μ pour la preuve que la méthode test-point n'est jamais plus faible que la méthode de séparation des enveloppes convexes pour $n = 3$

$$\nu_{003} = \lambda_{00300}$$
$$\nu_{012} = \lambda_{00201} + \lambda_{00210} + \lambda_{01200}$$
$$\nu_{021} = \lambda_{00102} + \lambda_{00111} + \lambda_{00120} + \lambda_{01101} + \lambda_{01110} + \lambda_{02100}$$
$$\nu_{030} = \lambda_{00003} + \lambda_{00012} + \lambda_{00021} + \lambda_{00030} + \lambda_{01002} + \lambda_{01011} + \lambda_{01020} + \lambda_{02001} + \lambda_{02010} + \lambda_{03000}$$
$$\nu_{102} = \lambda_{10200}$$
$$\nu_{111} = \lambda_{10101} + \lambda_{10110} + \lambda_{11100}$$
$$\nu_{120} = \lambda_{10002} + \lambda_{10011} + \lambda_{10020} + \lambda_{11001} + \lambda_{11010} + \lambda_{12000}$$
$$\nu_{201} = \lambda_{20100}$$
$$\nu_{210} = \lambda_{20001} + \lambda_{20010} + \lambda_{21000}$$
$$\nu_{300} = \lambda_{30000}$$

$$\mu_{003} = \lambda_{00030} + \lambda_{00120} + \lambda_{00210} + \lambda_{00300}$$
$$\mu_{012} = \lambda_{00021} + \lambda_{00111} + \lambda_{00201}$$
$$\mu_{021} = \lambda_{00012} + \lambda_{00102}$$
$$\mu_{030} = \lambda_{00003}$$
$$\mu_{102} = \lambda_{01020} + \lambda_{01110} + \lambda_{01200} + \lambda_{10020} + \lambda_{10110} + \lambda_{10200}$$
$$\mu_{111} = \lambda_{01011} + \lambda_{01101} + \lambda_{10011} + \lambda_{10101}$$
$$\mu_{120} = \lambda_{01002} + \lambda_{10002}$$
$$\mu_{201} = \lambda_{02010} + \lambda_{02100} + \lambda_{11010} + \lambda_{11100} + \lambda_{20010} + \lambda_{20100}$$
$$\mu_{210} = \lambda_{02001} + \lambda_{11001} + \lambda_{20001}$$
$$\mu_{300} = \lambda_{03000} + \lambda_{12000} + \lambda_{21000} + \lambda_{30000}.$$