DEMANDE D'HABILITATION

Dominique Michelucci
École Nationale Supérieure des Mines
158 cours Fauriel
42023 Saint-Étienne cedex 02

Table des matières

Chapitre 1

Introduction

La *modélisation géométrique* est une étape obligée pour de nombreux champs de l'informatique et de la simulation numérique, citons : la synthèse d'images, la reconnaissance des formes en analyse d'images, la robotique, ou la conception et la fabrication assistées par ordinateur (*CAO, CFAO*) ...

La modélisation géométrique est un de mes thèmes de recherche, en particulier la *modélisation géométrique par contraintes*. Cette dernière a des applications en synthèse d'images et en *CAO*. Elle permet à l'utilisateur de spécifier les contraintes géométriques que doivent satisfaire ses objets : par exemple, en $2D$, il pourra spécifier que tel cercle doit être tangent à telle droite, ou que tel angle doit avoir telle ou telle valeur. Idem en $3D$. La résolution des contraintes, et leur maintenance lors des modifications ou mises à jour, est à la charge du logiciel de modélisation. Pour l'utilisateur, la modélisation est donc simplifiée.

La modélisation géométrique par contraintes est l'objet du chapitre 2.

La programmation des méthodes géométriques se heurte à diverses difficultés, dont l'imprécision numérique. Les effets de cette dernière ont longtemps été sous estimés; aujourd'hui il est devenu patent que l'imprécision numérique peut gravement compromettre la crédibilité et la fiabilité des logiciels scientifiques. Aussi est-elle devenue un champ de recherches et de développements tant universitaires qu'industriels, qui pour être encore marginal n'en est pas moins vivace. On en veut pour preuves l'usage de plus en plus fréquent de diverses arithmétiques d'intervalles en analyse numérique (voir par exemple les numéros spéciaux de la revue *Computing* 53(3-4), en 1994) et même en synthèse d'images [66, 29], et le développement d'*ateliers de qualité numérique* par la société *Simulog* [1].

L'imprécision numérique dans les logiciels géométriques est mon deuxième thème de recherche. Le chapitre 3 lui est consacré. Plus généralement, ce chapitre traite des problèmes arithmétiques en géométrie; ainsi il présente aussi une arithmétique éliminant les cas particuliers en géométrie.

Le chapitre 4 présente les diverses thèses que j'ai co-encadrées, avec Mr Bernard Péroche.

Le chapitre 5 présente quelques activités annexes.

Chapitre 2

Modélisation par contraintes

2.1 Présentation

La *modélisation géométrique par contraintes* spécifie les objets géométriques (points, lignes, cercles ou autres coniques, courbes de Bézier ...en $2D$ et plans, tores, quadriques ...en $3D$) par des *contraintes*, typiquement : des distances ou des angles entre objets, des relations de parallélisme, d'incidence ou de tangence. Ce type de modélisation est aussi appelée *conception variationnelle* en CAO.

Ces contraintes donnent des systèmes d'équations, ou d'équations et d'inéquations, le plus souvent algébriques. Depuis les travaux fondateurs de Sutherland [68] et Borning [20], ces systèmes de contraintes ont été beaucoup étudiés. On peut classer grossièrement les méthodes de résolution en trois grandes catégories (non exclusives) :

- méthodes de *décomposition*, ou de *réduction*,
- méthodes *symboliques*, ou *formelles*,
- méthodes *numériques*.

Les méthodes de décomposition réduisent les systèmes de contraintes en d'autres systèmes plus simples, non décomposables : les irréductibles. Les systèmes irréductibles sont résolus séparément, puis les solutions partielles sont assemblées. En $2D$, des exemples typiques d'irréductibles sont les triangles : c'est à dire lorsque la position relative de trois sommets est définie par trois contraintes (par exemple 3 distances, ou 1 distance et 2 angles,...), ou bien les systèmes solubles à la règle et au compas comme le problème d'Appolonius [27]. Dans ces cas, une formule explicite permet de résoudre; cependant des sous-systèmes irréductibles plus complexes doivent être résolus par des méthodes symboliques ou numériques. La décomposition peut se faire soit par le moteur d'inférence d'un système expert, comme le proposent Buthion [24] ou Verroust *et al* [69], soit par un mécanisme d'unification fourni par un langage de programmation tel que Prolog, comme le propose Brüderlin [23], soit enfin par une méthode issue de la théorie des graphes, comme le proposent, entre autres, Owen [63], Bouma *et al* [21], ou Ait *et al* [4].

Les méthodes symboliques sont utilisées, par exemple, par Ericson et Yap [30] ou par Kondo [43]. Ces méthodes utilisent typiquement les bases de Gröbner, ou un genre de résultant, ou une autre technique d'élimination. Ces méthodes sont exactes mais particulièrement lentes : d'après Lazard [48] il est définitivement impossible de calculer les bases de Gröbner d'un système polynomial irréductible de degré 2 en 10 inconnues et équations. Ces méthodes semblent plus appropriées à la démonstration automatique de théorèmes, comme par exemple dans [28, 70], ou à la génération automatique de programmes de résolution numérique.

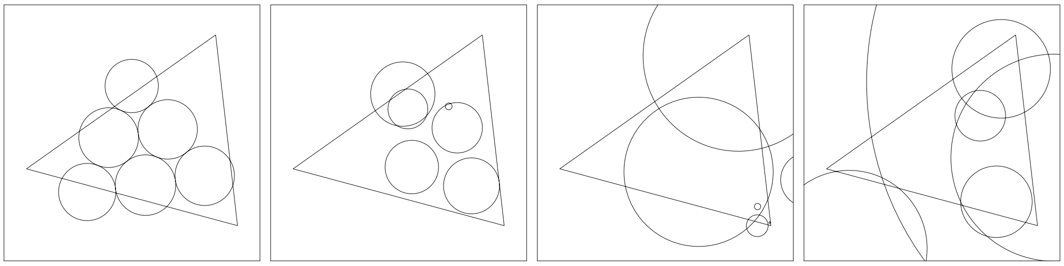

FIG. 2.1 - *Échec de la méthode de Newton-Raphson.*

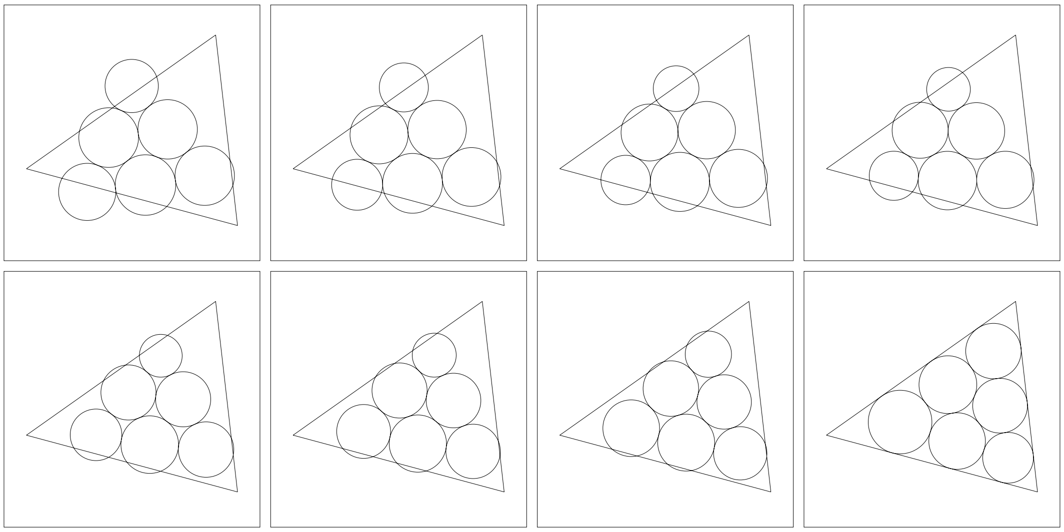

FIG. 2.2 - *Succès de l'homotopie.*

Les méthodes numériques sont donc incontournables pour résoudre les grands systèmes de contraintes que l'on rencontre dans le monde de la CAO. On pourra trouver des exemples d'utilisation chez Borning [20], Nelson [62], ou Light *et al* [50]. La méthode numérique généralement utilisée est celle de Newton-Raphson, ou une de ses variantes; elle nécessite une approximation initiale de la solution qui peut être fournie interactivement par l'utilisateur avec des outils tels que *MacDraw* ou *Xfig*. Dans une application interactive le besoin d'une approximation initiale n'est pas vraiment un inconvénient. Au contraire, elle permet de sélectionner naturellement la solution désirée parmi un nombre exponentiel de possibilités : d'après le théorème de Bezout, un système polynomial de degré total d, ayant n inconnues et équations possède $O(d^n)$ solutions. De plus la méthode de Newton-Raphson est rapide et fonctionne en temps polynomial. Enfin son utilisation est compatible avec l'utilisation de méthodes de décomposition (voir [4] pour un exemple) qui accélèrent la résolution.

2.2 Mon approche

J'ai étudié plus particulièrement deux questions en modélisation par contraintes : d'une part, la décomposition des systèmes de contraintes par des outils issus de la théorie des graphes ou de la théorie de la rigidité; et d'autre part la résolution de ces systèmes de contraintes.

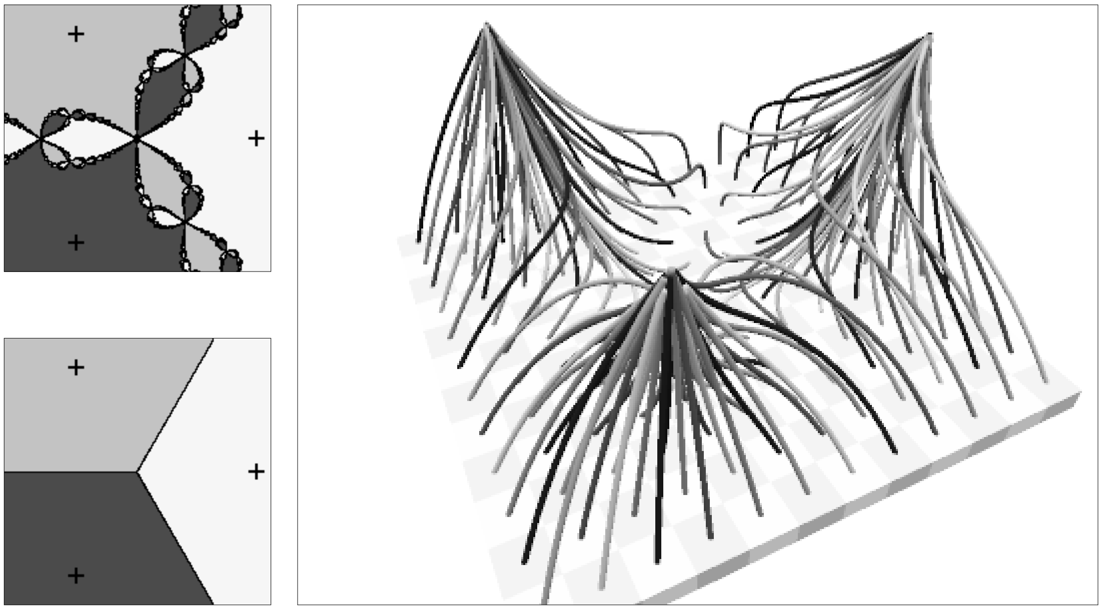

FIG. 2.3 - *Bassins d'attraction de $z^3 - 1 = 0$, $z \in C$ pour les méthodes de Newton (en haut à gauche) et l'homotopie (en bas à gauche). Courbes homotopiques en 3D (à droite).*

2.2.1 Résolution par homotopie

La méthode numérique la plus souvent utilisée pour résoudre les systèmes de contraintes est indéniablement celle de Newton-Raphson. On a vu qu'elle nécessite une approximation initiale de la solution, fournie interactivement par l'utilisateur. Cette méthode marche assez bien, mais parfois –trop souvent! – elle ne converge pas (fig. 2.1), ou converge vers une solution non désirée après un comportement erratique (du point de vue de l'utilisateur). Hervé Lamure et moi avons montré dans :

H. Lamure, D. Michelucci, Solving constraints systems by homotopy, 3^o ACM/IEEE Symposium on Solid Modeling and Applications, Salt Lake City, mai 1995. [47].

que l'*homotopie* a un comportement bien plus satisfaisant dans de tels cas (voir la figure 2.2, extraite de [47]) : elle est bien plus intuitive, et plus facilement prédictible que la méthode de Newton-Raphson. Ceci est dû au fait que les bassins d'attraction pour la méthode de Newton-Raphson sont des fractales (des ensembles de Julia, plus précisément), aux frontières très tourmentées, alors que ceux de la méthode homotopique sont des ensembles semi-algébriques (quand le système à résoudre est algébrique), aux frontières bien plus lisses : voir la figure 2.3, extraite de [47]. L'article [47] présente aussi une extension de l'homotopie, pour prendre en compte des systèmes sous-contraints : ceci permet à l'utilisateur d'ajouter contrainte par contrainte et de résoudre au fur et à mesure.

2.2.2 Décomposition

L'étude du graphe biparti d'incidence équations-inconnues, plus particulièrement la décomposition de Dulmage et Mendelsohn [51, 65], suffit souvent (dans les cas dits génériques) pour déterminer quelles inconnues sont sur-, sous-, ou bien-contraintes. Cette première décomposition est intéressante car elle permet de détecter des erreurs de l'utisateur dans les systèmes de contraintes. En effet, la programmation par contraintes reste de la programmation, et l'utilisateur a toujours la possibilité de commettre des erreurs...

Dans le cas bien contraint, une seconde décomposition, dite de König-Hall, permet de réduire les systèmes d'équations en systèmes irréductibles, ce qui accélère grandement la résolution, qu'elle soit numérique ou symbolique. A titre d'exemple, les temps de résolution d'un système (réductible) de 10 équations et inconnues sont divisés par environ 20 d'après nos expérimentations [3, 4].

Les décompositions de Dulmage-Mendelsohn et celle de König-Hall ne sont pas nouvelles mais sont très largement méconnues. En effet, il n'en existe pas d'exposé accessible; ainsi le livre de Berge [25], la référence obligée en théorie des graphes, n'en parle pas. À ma connaissance, les seuls livres traitant de ces questions sont celui de Lovasz et Plummer [51], celui de Recski [65], et celui de Murota [59]. Aussi l'usage de ces décompositions en analyse numérique, ou en modélisation par contraintes, est-il fort limité pour l'instant.

Ces décompositions sont présentées, avec quelques exemples d'utilisation, dans l'article :

S. Ait-Aoudia, R. Jégou, and D. Michelucci. Reduction of constraints systems. In Compugraphic*, pages 83–92, Alvor, Portugal, 1993.* [4]

Une variante de cet article a fait l'objet d'une communication lors de la conférence SEA93[1], qui a eu lieu au CIRM[2] à Marseille, en 1993. Cet article a aussi paru dans le rapport interne [41]. Il a été repris dans le deuxième chapitre de la thèse de S. Ait-Aoudia [3].

La décomposition de König-Hall s'avère insuffisante pour réduire complètement les systèmes de contraintes rencontrés en modélisation géométrique. Ces contraintes sont *indépendantes du repère*, pourvu que ce repère soit orthonormé : par exemple des relations de distance, d'angle, d'incidence ou de tangence entre objets. Même complètement déterminés, ces systèmes ne sont donc pas des systèmes bien-contraints, mais des systèmes sous-contraints très spéciaux, dits *rigides* : la position relative des objets géométriques (points, lignes, cercles...) est complètement déterminée, mais la position –et l'orientation– de l'ensemble ne l'est pas. En $2D$ (respectivement $3D$), il reste ainsi 3 (respectivement 6) degrés de liberté aux systèmes complètement déterminés. La décomposition de tels systèmes a été étudiée dans le cas bidimensionnel par Mr Hervé Lamure et moi même dans l'article :

H. Lamure, D. Michelucci, Decomposition of constraints systems, soumis à publication [45].

L'article [45] ne traite que le cas bidimensionnel. Il rappelle les bases de la théorie de la rigidité, puis présente deux méthodes de décomposition des systèmes de contraintes. La première décompose en rigides maximaux et détecte les cas de sur et sous déterminations; cette décomposition est incrémentale, autrement dit elle est mise à jour au fur et à mesure de la modification du système de contraintes; elle aide l'utilisateur à corriger les systèmes de contraintes erronés. La deuxième méthode s'applique aux systèmes bien déterminés, et les décompose en rigides élémentaires; elle permet d'accélérer la résolution : les parties élémentaires sont résolues indépendamment les unes des autres (éventuellement en parallèle), puis assemblées.

[1]SEA : Systémes d'Equations Algébriques.
[2]CIRM : Centre International de Recherches Mathématiques.

Chapitre 3

Problèmes arithmétiques en géométrie

3.1 Introduction

La programmation des méthodes de la géométrie algorithmique, voire des méthodes géométriques de la CAO [19] ou de la synthèse d'images, se heurtent à deux difficultés :

- l'imprécision numérique,
- le traitement des cas particuliers.

J'ai été confronté aux méfaits de l'imprécision numérique dès mon initiation à l'informatique graphique, lors de mon DEA et de mon travail de thèse [33, 53, 55]. J'ai recouru alors à une arithmétique exacte, et à quelques techniques ad hoc pour ne pas être pénalisé par la lenteur de cette arithmétique; voir 3.2.4. En 1993, j'ai animé un groupe (dit *groupe des paresseux*) constitué de deux étudiants alors en thèse : Mrs M.O. Benouamer et P. Jaillon, et de Mr J.M. Moreau, maître-assistant à l'EMSE[1]. Nous avons proposé la solution de l'*arithmétique paresseuse*. La section 3.2.1 résume nos travaux.

Bien que l'imprécision numérique ait existé dès la création des ordinateurs, l'imprécison numérique n'a pas été considérée d'emblée comme un sujet digne d'intérêt et de recherche dans les diverses communautés de l'informatique graphique : ce n'est qu'à la fin des années 1980 que ce thème a gagnè peu à peu droit de cité, avec un article de Greene et Yao [35] en 1986, l'article d'Hoffman *et al* [37] en 1988, l'article de Gangnet *et al* [32] et celui de Guibas *et al* [36] en 1989... Aujourd'hui encore, en géométrie algorithmique, il n'est pas rare que les opérations arithmétiques sur les nombres réels soient supposées être effectuées exactement et en temps constant.

La deuxième difficulté est le traitement des cas particuliers dans les algorithmes géométriques. Il y a peu de cas génériques, et leur traitement est relativement simple, en tout cas bien détaillé dans les algorithmes théoriques. Au contraire, il existe de nombreux cas particuliers; leur nombre croît exponentiellement avec la dimension spatiale, et leur traitement n'est presque jamais détaillé.

Il n'est pas évident que les cas particuliers constituent un problème de nature arithmétique, comme l'imprécision numérique. Telle est pourtant mon approche, que je résume en 3.3.

[1]EMSE : École Nationale Supérieure des Mines de Saint-Étienne

3.2 L'imprécision numérique en géométrie

3.2.1 Les méfaits de l'imprécision numérique

Les algorithmes de la géométrie algorithmique sont très sensibles aux imprécisions numériques inhérentes à l'emploi de l'arithmétique flottante. Non seulement ils peuvent donner des résultats aberrants, comme produire des structures de données représentant des topologies incohérentes, mais ils peuvent aussi ne pas arriver à leur terme. Ce phénomène a été constaté et a intéressé différents chercheurs, surtout depuis le début des années 1990 : citons Gangnet, Hoffman, Hopcroft, Iri, Karasick, Li, Lieber, Milenkovic, Nackmann, Sugihara, Yamagushi, Yap, et bien d'autres...

En effet, pour être efficaces, les méthodes de la géométrie algorithmique exploitent la transitivité de divers ordres, comme l'ordre lexicographique entre des sommets (on ordonne d'abord sur l'abscisse, puis sur l'ordonnée), ou l'ordre dans lequel une droite donnée coupe des arêtes. Ayant calculé que $a < b$ et que $b < c$, l'algorithme en déduit $a < c$ sans autre calcul. Il est donc impératif que la comparaison donne des résultats corrects, sinon, avec la propagation des résultats, une erreur numérique initialement minime se change en une erreur sur l'ordre des éléments, c'est à dire en une erreur de nature topologique, une incohérence.

Augmenter la précision des nombres flottants diminue la probabilité des erreurs lors des classements, mais ne l'annule pas : quelle que soit la longueur de la mantisse des nombres flottants utilisés, on peut trouver des cas de figures où l'arithmétique flottante ne peut ordonner les nombres de façon correcte; voici un exemple $2D$, qui suppose pour simplifier que l'arithmétique flottante est en base 10 sur 4 chiffres de mantisse (la généralisation est triviale). Considérez un sommet S de coordonnées exactes : $[6667/10000, 0]$ et exactement représenté en flottant par : $[0.6667, 0.0000]$; considérez ensuite deux segments s'intersectant exactement en $I = [2/3, 1]$, par exemple le segment "horizontal" entre les points $[0, 1]$ et $[1, 1]$, et le segment entre les points $[0, 0]$ et $[2, 3]$. Tous les nombres sont exactement représentés en flottant, à part l'abscisse du point I qui est au mieux : 0.6667. Ainsi, pour la machine, le point I est lexicographiquement supérieur au point S alors qu'en réalité, le point I est lexicographiquement inférieur au point S.

3.2.2 Quelques solutions proposées

Quelques solutions ont été proposées. L'heuristique des epsilon, où deux nombres différant de moins de epsilon sont décrétés égaux, est la solution la plus simple et la plus fréquemment utilisée (par exemple par Hughes, Laidlaw, et Trumbore [38]); elle fonctionne assez souvent..., mais pas toujours.

Guibas, Salesin et Stolfi [36] proposent d'utiliser des tests géométriques fiables qui rendent des valeurs : vrai, faux, ou indéterminé. Malheureusement, ils ne disent pas que faire en cas d'indétermination.

On peut remarquer que l'approche de Guibas, Salesin et Stolfi équivaut à l'utilisation d'une arithmétique d'intervalles. Malheureusement, une arithmétique d'intervalle ne permet pas de décider du signe d'un nombre, quand l'intervalle associé contient 0. Bien sûr, toutes les approches fondées uniquement sur l'emploi d'une arithmétique d'intervalle achoppent sur le test de nullité des nombres (et, de façon équivalente, de l'égalité de deux nombres).

Le *raisonnement symbolique* fait en sorte que l'algorithme prend toujours des décisions géométriques compatibles avec les décisions précédentes; ce principe simple est difficile à mettre en oeuvre, et peu d'algorithmes ont pu être reécrits pour s'y conformer. Cette approche a été proposée par Hoffman, Hopcroft et Karasick [37], par Li et Milenkovic [49], et par Milenkovic [57].

Une solution radicale est de s'interdire des algorithmes ou structures de données trop sensibles à l'imprécision numérique. Premier exemple : en CAO, le congrès CSG'94 a pour but d'étudier la possibilité de se passer de la structure de données dite *Représentation par les frontières* utilisée actuellement par la plupart des logiciels industriels de modélisation géométrique, et ce au profit d'une autre représentation a priori moins sensible à l'imprécision numérique. Deuxième exemple : les chercheurs du colloque *Imagerie discrète* proposent de renoncer carrément aux modèles géométriques continus.

L'arithmétique exacte est la méthode la plus évidente et la plus générale. Elle a malheureusement un coût en espace mémoire et en temps de calcul prohibitif, et ceci même dans le cas le plus favorable où une arithmétique rationnelle est suffisante. Pour ces raisons, plusieurs auteurs ont proposé l'utilisation d'arithmétiques mixtes, combinant calcul exact et calcul approximatif. Le principe est simple : les tests géométriques (typiquement le calcul du signe d'un produit scalaire ou d'un déterminant) sont effectués d'abord avec une arithmétique approchée, mais rapide, par exemple une arithmétique d'intervalles; c'est seulement si le résultat approché est ambigu (typiquement, l'intervalle contient zéro, et le signe est indéterminé) que l'on a recours, dans un deuxième temps, à l'arithmétique exacte. Diffèrentes arithmétiques approchées ont été utilisées, selon les chercheurs; citons Fortune et Van Wyk [31], Gangnet et Van Thong [34], Karasick, Lieber et Nackmann [42], Michelucci [33, 53], Moreau [58], Yamagushi et al [61] ...

Mais que faire lorsqu'il devient nécessaire de calculer et de stocker, disons, les coordonnées d'un point d'intersection entre deux segments ? Si les calculs ne sont effectués qu'en flottant (ce qui est bien sûr le plus rapide), l'information précise est perdue, alors qu'elle peut devenir nécessaire dans la suite des calculs. L'arithmétique mixte effectue donc en arithmétique exacte tous les calculs autres que les tests.

Certes, pour éviter le calcul exact, une alternative serait de conserver, pour chaque point d'intersection, des références aux deux segments : ainsi, si besoin est, il sera possible de calculer ultérieurement la valeur exacte des coordonnées. Ce type d'approche a été utilisé par Sugihara et Iri [67], et par Hoffman, Hopcroft et Karasick [26]. Bien sûr, il faut faire de même pour toutes les données générées en cours d'exécution, et la tâche du programmeur en est considérablement alourdie. L'arithmétique paresseuse automatise ce procédé.

3.2.3 L'arithmétique rationnelle paresseuse

L'arithmétique exacte paresseuse est inspirée par les deux remarques :

- un encadrement des valeurs numériques suffit le plus souvent.
- quand un calcul est demandé, on ne peut savoir si l'encadrement suffira.

Un nombre rationnel paresseux est représenté par :

- un intervalle entre deux nombres flottants, qui est garanti contenir le nombre, que la valeur exacte soit disponible ou non.
- une définition, qui est :
  - soit un nombre rationnel usuel (représenté par exemple par deux grands entiers, *ie* deux listes ou deux tableaux de chiffres dans une base donnée);
  - soit la somme (non évaluée) de deux autres nombres paresseux;
  - soit le produit (non évalué) de deux autres nombres paresseux;
  - soit l'inverse (non évalué) d'un autre nombre paresseux;
  - soit l'opposé (non évalué) d'un autre nombre paresseux.

Les nombres paresseux forment donc un arbre, ou plutôt un graphe orienté sans cycle car certains noeuds peuvent être partagés par plusieurs "pères". Effectuer une opération entre deux nombres paresseux revient à allouer une cellule, à calculer l'intervalle en fonction des intervalles des opérandes grâce à une arithmétique d'intervalle, classique; la valeur exacte n'est donc pas calculée, et ne le sera jamais si elle n'est jamais nécessaire. L'opération est effectuée en temps et espace constant.

Le calcul de la valeur exacte d'un nombre paresseux devient nécessaire quand :

- on veut le signe ou l'inverse d'un nombre dont l'encadrement contient 0;
- on veut comparer deux nombres dont les intervalles se chevauchent;
- on a besoin de la valeur exacte d'un père (ou d'un ancêtre) du nombre.

Le calcul de la valeur exacte d'un nombre rationnel paresseux utilise et "exécute" sa définition; à l'issue de l'évaluation, la définition arborescente est remplacée par la valeur exacte.

Nous avons programmé une librairie d'arithmétique rationnelle paresseuse. Elle a été testée avec succès sur trois programmes géométriques (calcul des points d'intersection entre segments dans le plan, par deux méthodes différentes, et calcul de l'intersection entre des polyèdres) et sur la résolution de systèmes linéaires par la méthode de Gauss. Cette librairie est considérablement plus rapide que l'arithmétique rationnelle : 100 fois plus rapide, voire plus de 1000 fois. Cette librairie est cependant environ 10 ou 20 fois plus lente que l'arithmétique flottante (quand les programmes utilisant l'arithmétique flottante arrive à leur terme)[2].

Certains algorithmes géométriques utilisent des tables à adressage dispersé (*hash table*) pour retrouver, disons, les sommets en fonction de leurs coordonnées, qui sont donc utilisées comme des clés. L'utilisateur d'une arithmétique paresseuse est alors confronté à un problème : il ne peut calculer cette clé, car la valeur numérique exacte des nombres paresseux n'est pas disponible, en règle générale. Comment calculer une clé sans évaluer le nombre?

En se fondant sur l'arithmétique modulaire, il est possible de calculer une telle clé. La clé d'un rationnel dont la représentation irréductible a/b est disponible est ab^{-1} $moduloP$, où P est un nombre premier, typiquement $2^{31}-1$, et où b^{-1} est l'inverse de b $moduloP$; quand $b = 0$ $moduloP$, la clé est Ω, où Ω est un nombre hors de $[0, P[$, par exemple P. Quant à la clé d'un nombre paresseux dont la valeur rationnelle n'est pas disponible, elle peut se déduire des clés des nombres intervenant dans la définition, sauf (rares) cas d'indétermination. En supposant que X et X' ont respectivement pour clé x et x' :

- le nombre $X + X'$ a pour clé $x + x'$ $moduloP$ quand $(x, x') \neq (\Omega, \Omega)$
- le nombre XX' a pour clé xx' $moduloP$ quand $(x, x') \neq (0, \Omega)$ et $(x, x') \neq (\Omega, 0)$
- le nombre $-X$ a pour clé $-x$ $moduloP$, avec $-\Omega = \Omega$
- le nombre $1/X$ a pour clé x^{-1} $moduloP$, avec $1/0 = \Omega$ et $1/\Omega = 0$.

Actuellement, l'arithmétique paresseuse recourt à l'évaluation lors de la comparaison de nombres ayant des définitions non identiques mais algébriquement équivalentes; une programmation *avertie* de la part de l'utilisateur résout cette difficulté mais est peu naturelle [40]. Quelques autres questions ouvertes : l'utilisation de l'arithmétique paresseuse en analyse numérique avec des algorithmes itératifs? la possibilité d'une arithmétique paresseuse algébrique?

[2]En fait, les performances de l'arithmétique paresseuse ont été améliorées depuis, par Mr P. Jaillon puis par Mr J.M. Moreau et un étudiant en stage. Cependant des mesures systématiques de performance n'ont pas encore été effectuées.

3.2.4 Articles publiés sur ce thème

Construction de cartes planaires :

M. Gangnet, D. Michelucci, Un outil graphique interactif, MICAD 84, *Hermès, 95–110.* [33]

Le logiciel décrit dans cet article est une aide à la saisie d'esquisses d'architectes. Il échantillonne des tracés à main levée. Il reconnait des segments, les recale selon des directions privilégiés, et construit une *carte planaire* du dessin. Une carte planaire est une structure de données représentant des zones et leurs frontières dans le plan, et les relations topologiques de contiguïté et d'inclusion entre les différentes régions. La carte planaire permet ensuite différentes manipulations : effacement des arêtes pendantes (qui sont souvent, dans ce contexte, des imperfections du tracé manuel), désignation d'arcs ou de régions et attachement d'attributs (épaisseur, texture, nom...). Les points d'intersection entre les tracés sont déterminés grâce à la méthode de Bentley-Ottman. L'article distingue carte planaire *locale* et *globale*, et présente un algorithme en temps linéaire de construction de la carte globale à partir de la carte locale.

Les problèmes d'imprécision sont signalés (déjà...), et résolus par l'emploi d'une arithmétique exacte rationnelle de dynamique limitée : les coordonnées des points initiaux sont arrondis sur une grille régulière à coordonnées entières, entre 0 et 2.10^4.

Remarque : à l'époque, j'étais étudiant en architecture à Saint-Étienne, et cet article présente mon travail de stage de 4^o année réalisé à l'EMSE[3] sous la direction de Mr Michel Gangnet. Cet article m'a permis de m'inscrire sur dossier au DEA de synthèse d'images, DEA cohabilité par l'Université et par l'EMSE. Ce travail fait aussi l'objet du premier chapitre de ma thèse d'informatique [53].

Certaines des structures de données ou des méthodes de cet article ont été reprises par Mrs H. Ben Amara [5] et A. Nahed [60], pour leur thèse, et pour un contrat européen EspritII : MMI^2 (*Multi-Mode Interface for Man-Machine Interaction*). Voir aussi le rapport interne [6]. Ce travail a aussi été utilisé par un outil graphique interactif réalisé par Mr Michel Gangnet *et al* [32].

Perspectives d'arbres de construction :

D. Michelucci, B. Péroche, Représentation interactive d'arbres CSG, In MICAD 87, *Hermès, 535–548.* [55]. Voir aussi le rapport interne [56].

L'arbre de construction, ou arbre CSG (*Constructive Solid Geometry*) représente, typiquement, les objets de l'architecture ou de la mécanique. Ses feuilles sont des solides élémentaires (cubes, cônes, cylindres, sphères, tores...) et ses noeuds sont des opérations booléennes (intersection, différence, union) ou des transformations affines (translations, rotations, affinités, symétries). Cette représentation ne fournit pas explicitement les frontières des objets. Cet article propose une méthode qui donne les frontières, en fait la carte planaire, d'une perspective centrale ou axonométrique d'une scène décrite par arbre de construction. Les solides élémentaires de la scène sont d'abord facétisés. La carte planaire de la projection de toutes leurs arêtes est calculée. Dans chaque zone de cette carte, les solides élémentaires présents sont déterminés, et les faces sont ordonnées par distance croissante à l'oeil. Si besoin est, certaines zones sont coupées par des arêtes d'intersection entre les faces des solides élémentaires. Dans chaque zone, la face vue est calculée. Les arêtes vues ou cachées s'en déduisent facilement. Finalement, la carte planaire de la perspective est obtenue.

Cet article discute et traite les problèmes dus à l'imprécision numérique.

[3] EMSE : École Nationale Supérieure des Mines de Saint-Étienne.

Ce travail a fait l'objet du deuxième chapitre de ma thèse [53]. Avec le recul, on peut remarquer, un peu perfidement, que calculer une carte planaire d'une perspective d'un arbre de construction est un problème aussi difficile et aussi coûteux en temps d'exécution que le calcul effectif des frontières d'un arbre de construction (voir 3.2.4). Il vaut donc mieux résoudre ce deuxième problème, et utiliser ensuite une méthode classique pour calculer la carte planaire de la perspective d'un objet dont les frontières sont connues. Mme A. Verroust, qui a elle aussi proposé une méthode de visualisation des arbres de construction, a fait le même constat. Malgré tout, ce travail n'a pas été du temps perdu car il m'a beaucoup appris.

Opérations booléennes entre polyèdres :

M.O. Benouamer, D. Michelucci, and B. Péroche, Boundary evaluation using a lazy rational arithmetic, In Proceedings of the 2^o ACM/IEEE Symposium on Solid Modeling and Applications*, pages 115–126, Montréal, Canada, 1993.* [18]

Une version revue et corrigée de cet article a ensuite été publiée dans :

M.O. Benouamer, D. Michelucci, and B. Péroche, Error-free boundary evaluation based on a lazy rational arithmetic: a detailed implementation, In Computer-Aided Design*, pages 403–416, Volume 26, Number 6, June 1994.* [19]

Ce travail a aussi fait l'objet du rapport interne [17]. L'article présente une méthode de calcul des frontières de l'intersection, de la réunion ou de la différence entre deux polyèdres représentés par leurs frontières. Cet article propose une structure de données simplifiant et unifiant les très nombreux cas particuliers rencontrés dans ce problème. De plus, c'est le premier article exposant le principe de l'arithmétique rationnelle paresseuse (voir 3.2.4).

L'algorithme et les structures de données font l'objet du 3^o chapitre de ma thèse d'informatique [53]. Ils ont été effectivement programmés par Mr M.O. Benouamer lors de son travail de thèse [10]. La première implantation etait très ralentie par l'emploi d'une arithmétique exacte rationnelle. C'est à la suite de ce constat que nous avons introduit l'arithmétique rationnelle paresseuse.

Solid Modeling est une conférence internationale sur la modélisation des solides, qui a lieu tous les deux ans. Les meilleurs articles sont ensuite republiés dans des numéros spéciaux (*special issues*) de revues telles que *Computer-Aided Design*, ou *Computer Graphics.*

Imprécision et arithmétique paresseuse :

M.O. Benouamer, P. Jaillon, D. Michelucci, and J-M. Moreau. A lazy solution to imprecision in computational geometry, In Proceedings of the 5^o Canadian Conference on Computational Geometry*, pages 73–78, Waterloo, Canada, August 5-9, 1993.* [11]

L'imprécision numérique provoque de graves incohérences, notamment topologiques, dans les algorithmes de la géométrie algorithmique. Cet article donne des exemples d'incohérence, et rappelle les solutions précédemment proposées. La solution théorique la plus simple et la plus générale est l'arithmétique exacte rationnelle; en pratique, elle est malheureusement inutilisable car beaucoup trop lente. L'article présente alors l'arithmétique exacte paresseuse, résumée en 3.2.1. Faute de place, cet article ne donne que les principes de l'arithmétique paresseuse.

Arithmétique paresseuse rationnelle :

M.O. Benouamer, P. Jaillon, D. Michelucci, and J-M. Moreau. A lazy arithmetic library, In Proceedings of the IEEE 11th Symposium on Computer Arithmetic, *Windsor, Ontario, June 30-July 2, 1993.* [12]

Cet article détaille l'arithmétique rationnelle paresseuse. L'étiquetage des nombres paresseux n'est pas abordé (voir 3.2.4). Faute de place, cet article ne discute que très sommairement les méfaits de l'imprécision numérique sur les programmes de géométrie algorithmique, et les solutions antérieures : ces points ne sont pas les plus intéressants pour l'audience de *Computer Arithmetic*. *Computer Arithmetic* est la conférence internationale sur l'arithmétique des ordinateurs, et a lieu tous les deux ans. Une version un peu plus détaillée de cet article a été publiée en rapport interne [16].

Étiquetage des nombres paresseux :

M.O. Benouamer, P. Jaillon, D. Michelucci, and J-M. Moreau, Hashing lazy numbers, In Proceedings of SCAN-93, *Vienna, Austria, September 1993.* [13]

Une variante de cet article a ensuite été publiée dans la revue *Computing* [14]. Ce travail a fait l'objet du rapport interne [15]. Le problème de l'étiquetage des nombres paresseux s'est posé lors de la programmation de la méthode exposée en 3.2.4. L'étiquette fait naturellement envisagé des arithmétiques exactes probabilistes (voir 3.2.5).

Article soumis à publication :

Cet article très complet fait le point sur l'arithmétique exacte rationnelle paresseuse :

D. Michelucci, J.M. Moreau, Lazy Arithmetic, soumis à publication. [54]

3.2.5 Recherches futures

Arithmétique rationnelle probabiliste :

Les travaux sur l'arithmétique paresseuse m'ont fait envisager une arithmétique exacte probabiliste. Les expérimentations avec l'arithmétique paresseuse ont montré que tous les nombres paresseux dont la clé est nulle et dont l'intervalle contient 0 et est de faible amplitude (disons 10^{-6}) sont vraiment nuls. D'où l'idée de prendre le risque de déclarer nuls de tels nombres, sans calculer (par l'arithmétique rationnelle) leur valeur exacte. L'étude, expérimentale et théorique, de cette arithmétique probabiliste devrait être conduite avec Jean-Michel Moreau.

Arithmétique quadratique :

Par jeu, j'ai programmé une *arithmétique quadratique*, permettant d'effectuer de façon exacte l'extraction de la racine carrée, en sus des opérations arithmétiques usuelles : $+, -, \times, /$, $\leq$, $=$. L'idée consiste à calculer dans des extensions successives de Q. Elle a fait l'objet d'un projet de programmation en *Scheme*, pour des élèves de première année de l'EMSE, pendant l'année 1993-1994. On trouvera l'énoncé et le principe de la solution en appendice. Au congrès *"Les nombres réels et l'ordinateur"* (EMSE, 4-6 avril 1995), Denis Bouhineau a récemment décrit une arithmétique quadratique similaire, programmée en *Prolog3*. Mon implantation, en Lelisp, n'est pas paresseuse : je trouve le champ d'application de l'arithmétique quadratique trop restreint.

Arithmétique algébrique :

Est-il possible de généraliser la paresse aux arithmétiques exactes algébriques? J'en ai discuté avec Dominique Duval, de l'Université de Limoges. Elle est un des auteurs de la représentation D_5 des nombres algébriques[4]. Il est relativement simple d'appliquer le principe de la paresse à la représentation D_5, ce qui, *a priori* devrait fournir une arithmétique algébrique plus efficace. Cependant, je crains qu'une telle arithmétique soit encore trop lente pour les besoins de la CAO ou de la géométrie algorithmique (l'arithmétique rationnelle paresseuse, elle, est presque utilisable industriellement). Fondamentalement, cela vient de ce que les techniques d'élimination (résultants, bases de Gröbner, pgcd dans D_5, etc) ont un coût exponentiel ou doublement exponentiel.

Une autre piste envisageable est l'arithmétique algébrique probabiliste. Mais l'arithmétique rationnelle probabiliste est-elle généralisable aux cas algébrique? Autrement dit peut-on trouver des clefs, des signatures, pour les nombres algébriques?

3.3 La suppression des cas particuliers

Le traitement des cas particuliers est une tâche fort pénible quand on s'avise de programmer effectivement les méthodes de la géométrie algorithique. En $2D$, quelques cas particuliers typiques sont : l'alignement de plus de trois points, le parallélisme ou la confusion entre deux droites, la cocircularité de plus de trois points, un point commun à plus de deux droites, plus de deux sommets ayant même abscisse ou même ordonnée... Le nombre de cas particuliers croît exponentiellement avec la dimension de l'espace. Mathématiquement, il y a cas particulier quand le signe d'une expression (un déterminant, typiquement) dans un test géométrique s'avère nul, les seuls signes dits *génériques* étant positif et négatif. La *simulation de la simplicité* propose de perturber symboliquement les données par des grandeurs infinitésimales. Ces grandeurs infinitésimales ne sont pas explicitement représentées en machine. En pratique, les librairies actuelles simulant la simplicité (symptomatiquement appelées : boites noires) ont des fonctionnalités limitées (par exemple à des calculs de déterminants), et sont d'une utilisation fort pénible; ainsi il est difficile de spécifier le signe d'une perturbation. Il est impossible d'utiliser ces librairies dans des algorithmes en-ligne.

Je propose une nouvelle approche, qui supprime ces limitations: les grandeurs infinitésimales sont explicitement représentées en machine; l'article [52] suivant :

D. Michelucci, An ϵ-arithmetic for removing degeneracies, soumis à publication.

présente une arithmétique capable de calculer sur des infinitésimaux (ou des infiniment grands). Une fois que le programmeur a perturbé les données, *c'est l'arithmétique elle-même qui élimine les cas particuliers*: le programmeur n'a plus à considérer les cas dégénérés, mais seulement les cas génériques.

L'article propose deux arithmétiques. Dans la première, une donnée p est perturbée en $P = p + p_1\epsilon + p_2\epsilon^2 + \ldots$: le développement de P contient une quantité (potentiellement) infinie de termes aléatoires p_i; il n'y a qu'un seul ϵ. Dans la deuxième arithmétique, la donnée p_i est perturbée par $p_i + \epsilon_i$, et il faut ordonner de façon cohérente tous les monômes en ϵ_i. L'article discute et compare ces deux arithmétiques.

Enfin, pour certains algorithmes, le programmeur souhaite exercer un contrôle plus fin sur la perturbation. Par exemple il veut contrôler le signe de certaines perturbations, ou veut perturber un point de telle façon qu'il reste sur une demi droite donnée : toutes choses impossibles, ou extrêmement compliquées avec les perturbations précédemment proposées. Avec mon approche, les infinitésimaux sont des nombres "comme les autres", et ce type de contrôle devient trivial.

[4] D_5 est un raccourci pour : Dominique Duval, Dicrescenzo, Della Dora.

Chapitre 4

Thèses encadrées

J'ai été membre des jurys des thèses de Mrs M.O. Benouamer [10], P. Jaillon [39], S. Ait-Aoudia [3] qui ont eu lieu à l'EMSE; et des thèses de Mrs J.C. Bajard [7] et S. Kla [44] qui ont eu lieu à l'ENS[1] de Lyon.

Cette partie liste les étudiants en thèse à l'EMSE que j'ai encadrés, ou encadre actuellement. Dans tous les cas, le directeur de recherche est Mr Bernard Péroche, professeur de l'EMSE.

4.1 Le groupe des paresseux

J'ai animé en 1993 un groupe travaillant sur l'arithmétique rationnelle paresseuse, constitué de deux étudiants alors en thèse : Mrs M.O. Benouamer et P. Jaillon, de Mr J.M. Moreau, maître-assistant à l'EMSE, et de moi-même. La librairie d'arithmétique paresseuse a été programmée principalement par Mr P. Jaillon et moi. Les programmes géométriques qui ont servi de banc d'essai à cette librairie ont été programmés par Mr M.O. Benouamer et moi.

4.2 Thèse de Mr M.O. Benouamer

M.O. Benouamer, Operations booléennes sur les polyèdres représentés par leurs frontières et imprécisions numériques*, thèse, EMSE, 1993.*

La thèse de M.O. Benouamer traite des opérations booléennes entre polyèdres. Les deux difficultés majeures de ce problème sont l'imprécision numérique, qui provoque des incohérences, et les très nombreux cas particuliers. Le programme de Mr M.O. Benouamer a aussi servi de banc d'essai pour l'arithmétique paresseuse. Voir 3.2.4 et 3.2.4. Ensemble, nous avons publié les articles [11, 12, 13, 14, 18, 19].

4.3 Thèse de Mr P. Jaillon

P. Jaillon, Proposition d'une arithmétique rationnelle paresseuse et d'un outil d'aide à la saisie d'objets en synthèse d'images*, thèse, EMSE, 1993.*

La première moitié du mémoire de thèse de Mr P. Jaillon présente l'implantation de la librairie d'arithmétique rationnelle paresseuse, et quelques expérimentations. Ensemble, nous avons publié les articles [11, 12, 13, 14].

[1]ENS : École Normale Supérieure.

4.4 Thèse de Mr S. Ait-Aoudia

S. Ait-Aoudia, Modélisation géométrique par contraintes : quelques méthodes de résolution*, thèse, EMSE, 1994.*

Ensemble, nous avons publié l'article [4]. S. Ait-Aoudia a aussi présenté une contribution aux journées de l'AFIG [2].

4.5 Thèse de Mr H. Lamure

La thèse de H. Lamure est en cours. Elle porte sur la modélisation géométrique par contraintes. Ensemble nous avons publié l'article [47] : voir 2.1; un autre article est soumis à publication : voir 2.2.2. Mr H. Lamure a aussi présenté ces travaux aux journées 1994 de l'AFIG à Toulouse [46].

Chapitre 5

Activités diverses

5.1 Quelques articles ou conférences

L'article suivant :

M. Beigbeder, D. Michelucci and J.M. Moreau, Object-Oriented Languages and Generalized Polyedra, In Proceedings of third Eurographics Workshop on Object-Oriented Graphics, Champéry, Switzerland, October 1992 [9]. Voir aussi le rapport [8].

discute les intérêts et les limitations des langages orienté-objet pour la programmation des méthodes géométriques, en particulier la manipulation des polyèdres ou la redéfinitions des opérations arithmétiques.

Finalement, voici quelques interventions récentes, dans diverses conférences françaises sans comité de lecture :

B. Péroche, D. Michelucci, La modélisation géométrique par contraintes : tour d'horizon des méthodes de résolution, Journées 3IA*, Limoges, 1994.*

D. Michelucci, L'arithmétique paresseuse, pages 69–78, Actes des Journées AFIG[1] 1993, Bordeaux.

H. Lamure, D. Michelucci, Résolution de contraintes par une méthode homotopique, Journées AFIG 94*, Toulouse (une variante française de l'article décrit en 2.1). [46]*

5.2 Organisation de conférences

À l'instigation de Mr Jean-Michel Muller (CNRS, Laboratoire LIP, ENS de Lyon), une conférence est organisée les 4, 5, et 6 avril 1995 à l'EMSE. Elle porte sur *"Les nombres réels et l'ordinateur"*. Le but est de faire se rencontrer les diverses communautés scientifiques (analyse numérique, calcul formel, géométrie algorithmique, CAO...) confrontées aux limitations des arithmétiques disponibles sur les ordinateurs actuels : problèmes de précision, de fiabilité, de portabilité, de rapidité, de standardisation... Le comité d'organisation comprend : Mrs Jean-Claude Bajard (Laboratoire LIM, Université de Provence), Hamid Bessalah (Centre de développement des technologies avancées, Alger), Marc Daumas (Laboratoire LIP, ENS de Lyon), Xavier

[1] AFIG : Association Française de l'Informatique Graphique. Les Journées AFIG sont l'occasion pour les jeunes chercheurs en thèse de présenter leurs travaux à la communauté de l'informatique graphique française.

Merrheim (Laboratoire LIP, ENS de Lyon), Mme Anne Mignotte (Laboratoire LIP, ENS de Lyon), Mrs Jean-Michel Moreau (EMSE), Jean-Michel Muller (CNRS, Laboratoire LIP, ENS de Lyon), et moi. On trouvera en appendice l'appel à communication.

J'ai fait partie de deux comités de lecture : celui d'IMARA 93 : les rencontres architecturales lors d'IMAGINA en 1993, et la conférence *"Les nombres réels et l'ordinateur"* en avril 1995.

5.3 Enseignement

L'enseignement ne représente que 10 à 15% de mon activité. Je liste ici les enseignements effectués ces deux dernières années.

5.3.1 À l'École des Mines de Saint-Étienne

En 1993-1994 et 1994-1995, avec Mrs Roland Jégou et Jean-Michel Moreau, j'ai donné les cours de Géométrie Algorithmique lors du DEA d'Analyse et de Synthèse d'Images; ce DEA est cohabilité par l'Université de St-Étienne et l'EMSE. J'ai encadré les séances de TD-TP de programmation en langages fonctionnels (*Scheme*) et des projets d'étudiants (quelques exemples typiques de sujet sont donnés en appendice) en 1ère année; les cours étaient assurés par Mr Jean-Jacques Girardot. En 1994-1995, j'ai encadré les TD-TP de programmation en C++ en 1ère année; les cours étaient assurés par Mr Philippe jaillon. J'ai encadré les projets d'élèves en deuxième année, qui suivaient les cours *"Concepts UNIX/C et Algorithmique"* assurés principalement par Mrs Michel Beigbeder, Marc Roelens et Roland Jégou.

Pour finir, je citerai quelques interventions ponctuelles : encadrement d'élèves-ingénieurs lors de leur constitution d'un dossier bibliographique sur, par exemple, la théorie du chaos (fractals, attracteurs étranges), l'utilisation des réseaux neuromimétiques en reconnaissance des formes, etc; participation aux jurys lors des soutenances d'élèves (dossier bibliographique, stage en entreprise, projet de programmation).

5.3.2 Autres cours

Depuis 1983, je fais chaque année un cours d'introduction à la synthèse d'images et à la modélisation géométrique aux étudiants du CEA (Certificat d'Etudes Archtecturales) de l'École d'Architecture de Marseille-Luminy. Ce cours dure une trentaine d'heures et est fait conjointement avec une autre personne de l'Equipe de Synthèse d'Images de l'EMSE (Mr Bernard Péroche, Mlle Sabine Coquillart, Mr Michel Beigbeder, Mr Philippe Jaillon, Mr Gilles Mathieu, selon les années). Les étudiants sont, typiquement, de jeunes architectes qui souhaitent acquérir une initiation à l'informatique, essentiellement en tant qu'utilisateur des logiciels de CAO. L'organisation de ce CEA est assurée par le GAMSAU[2]; incidemment, c'est au sein du GAMSAU que j'ai pasé mon diplôme d'architecte DPLG.

J'ai effectué d'autres cours d'introduction à la synthèse d'images, à la modélisation géométrique et à la géométrie algorithmique pendant les années scolaires 1991-1992, 1992-1993, 1993-1994 à l'ENS de Lyon, au sein du DEA d'Informatique Fondamentale. Ce cours a été effectué conjointement avec Mr Bernard Péroche.

5.4 Livres

Mr B. Péroche, Mme J. Argence, Mr D. Ghazanfarpour et moi avons rédigé le livre d'introduction à la synthèse d'images : *"La synthèse d'images"*, chez Hermès, en 1988 [64]. C'est l'un des premiers livres en

[2]GAMSAU : Groupe d'Application des Méthodes Scientifiques à l'Architecture et à l'Urbanisme

français sur ce sujet. Je m'étais plus particulièrement chargé des méthodes $2D$ (à l'exception des problèmes d'aliassage, traités par Mr D. Ghazanfarpour), de la modélisation, et des méthodes d'élimination des parties cachées.

Une nouvelle version est prévue, Mr Roelens remplaçant Mme Argence. Pour l'instant, avec Mr D. Ghazanfarpour, nous avons –en gros– achevé la rédaction du chapitre traitant des méthodes $2D$ (35 pages). Plus globalement, un peu moins du tiers de l'ouvrage a été rédigé, pour l'instant.

5.5 Programmation, production

En 1987 et 1988, à l'EMSE, j'ai développé un petit logiciel de modélisation (*castorC*) pour la synthèse d'images, et programmé un algorithme de visualisation, grâce auxquels j'ai réalisé un film d'animation en synthèse d'images, avec Mrs Michel Beigbeder, Frank Chopin et quelques autres personnes. Malgré ses limitations, notamment l'abscence d'interactivité, ce modeleur a été utilisé par beaucoup des chercheurs en thèse de l'équipe de synthèse d'images; le dernier en date est Mme Véronique Bourgoin [22], qui a réalisé un petit film d'animation en images de synthèse, avec l'aide de Mr Marc Roelens. Je suis actuellement en train de reprendre ce modeleur, pour qu'il soit utilisable depuis des applications interactives, et plus généralement pour qu'il permette aux chercheurs en thèse de tester rapidemment leurs idées (par exemple, la modélisation géométrique par contraintes, en $3D$).

En 1992, avec Mrs Michel Beigbeder, Philippe Jaillon et Marc Roelens, nous avons aussi réalisé un autre petit film de synthèse sur la fécondation in vitro.

Curriculum Vitæ

Dominique (Jean) Michelucci

- Né le 6 janvier 1959 à Firminy (42).
- Français. Célibataire. Sans enfant.
- Adresse personnelle: 19, rue de l'Industrie, 42240 Unieux.
- Téléphone personnel: (33).77.56.20.52.
- Ingénieur de recherche.
- Adresse professionnelle: École des Mines, 158 cours Fauriel, 42023 Saint-Étienne cedex 2.
- Téléphone professionnel: (33).77.42.01.73.
- Adresse électronique: *micheluc@emse.fr*

Titres

- 1977. Bacc C (mention B.).
- 1985. DEA d'Informatique. *Analyse et synthèse d'image, intelligence artificielle.* Université et École des Mines de St-Étienne (mention B.).
- 1987. Doctorat en Informatique. Université et École des Mines de St-Étienne (mention T.H.).
- 1989. Architecte DPLG. École d'Architecture de Marseille-Luminy (félicitations du jury).

Emplois

- 1985–1987. Boursier à l'École des Mines de St-Étienne.
- 1987–1988. Attaché de recherche à l'École des Mines de St-Étienne.
- 1989. Ingénieur chez SOGITEC à Boulogne.
- 1990–1995. Ingénieur de recherche à l'École des Mines de St-Étienne.

Dernières activités d'enseignement

- TP-TD, encadrement de projets, en: *Scheme*, *C* ou *C++*, *UNIX*, algorithmique, dossier bibliographique, à l'École des Mines de St-Étienne, 1993–1994 et 1994–1995.
- Cours de géométrie algorithmique en DEA d'informatique à l'École des Mines de St-Étienne, 1990–1995.
- Cours d'introduction à la synthèse d'image et à la géométrie algorithmique, au DEA de l'École Normale Supérieure de Lyon, 1991–1994.
- Idem au CEA de l'École d'architecture de Marseille-Luminy, 1990–1995.
- Thèses co-encadrées avec Mr Péroche, à l'École des Mines de St-Étienne :
  - S.A. Aoudia, thèse soutenue en 1994. 1 publication.
  - M.O Benouamer, thèse soutenue en 1993. 6 publications.
  - P. Jaillon, thèse soutenue en 1993. 4 publications.
  - H. Lamure, thèse en cours. 1 publication.

Collaboration avec d'autres équipes

- Le groupe *Calcul Formel/Calcul Numérique* : Université de Limoges, et J.P. Sabatier de Toulouse.
- Le groupe PRC *Arithmétique des Ordinateurs*, animé par J.M. Muller du LIP (ENS Lyon).

Principales publications

- S. Ait-Aoudia, R. Jegou, and D. Michelucci. Reduction Of Constraint Systems. In *Compugraphic*, pages 83–92, Alvor, Portugal, 1993.
- J. Argence, D. Ghazanfarpour, D. Michelucci et B. Péroche. La synthèse d'images. Hermès, 1988.
- M.O. Benouamer, P. Jaillon, D. Michelucci, and J-M. Moreau. A Lazy Solution To Imprecision In Computational Geometry. In *Proceedings of the 5th Canadian Conference on Computational Geometry*, pages 73–78, Waterloo, Canada, August 5-9, 1993.
- M.O. Benouamer, P. Jaillon, D. Michelucci, and J-M. Moreau. A Lazy Arithmetic Library. In *Proceedings of the IEEE 11th Symposium on Computer Arithmetic*, Windsor, Ontario, June 30-July 2, 1993.
- M.O. Benouamer, P. Jaillon, D. Michelucci, and J.M. Moreau. Hashing Lazy Numbers. *Computing*, 53(3-4):205–217, 1994.
- M.O. Benouamer, D. Michelucci, and B. Péroche. Error-Free Boundary Evaluation Based On A Lazy Rational Arithmetic: A Detailed Implementation. *Computer-Aided Design*, 26(6):403–416, June 1994.
- M. Gangnet and D. Michelucci. Un outil graphique interactif. *actes de MICAD 84*, pages 95–110. Hermès, 1984.
- H. Lamure and D. Michelucci. Solving Constraints Systems by Homotopy. In *Proceedings of* 3° *ACM/IEEE Symposium on Solid Modeling and Applications*, May 1995.
- D. Michelucci and B. Péroche. Représentation interactive d'arbres *CSG. Actes de MICAD 87*, pages 535–548. Hermès, 1987.

Bibliographie

[1] Aquarels: Atelier de qualité numérique pour la réalisation de logiciels scientifiques. Technical Report RAP.TS.94.sgi.020, SIMULOG, 1994.

[2] S. Ait-Aoudia. Modélisation 3d à l'aide de contraintes géométriques. In *Actes des Journées GROPLAN 92, Nantes*, 1992.

[3] S. Ait-Aoudia. *Modélisation géométrique par contraintes: quelques méthodes de résolution*. PhD thesis, École Nationale Supérieure des Mines de Saint-Étienne, 1994.

[4] S. Ait-Aoudia, R. Jegou, and D. Michelucci. Reduction of constraint systems. In *Compugraphic*, pages 83–92, Alvor, Portugal, 1993.

[5] H. Ben Amara. *Un outil graphique pour une interface multi-modes*. PhD thesis, Ecole des Mines, St Etienne, France, 1992.

[6] H. Ben Amara, D. Michelucci, and B. Péroche. A graphic interface for mmi2. Technical Report 90.5, Ecole des Mines de St-Etienne, February 1990.

[7] J.C. Bajard. *Évaluation de fonctions dans des systèmes redondants d'écriture des nombres*. PhD thesis, École Normale Supérieure de Lyon et Université Claude Bernard Lyon 1, 1993.

[8] M. Beigbeder, D. Michelucci, and J.M. Moreau. Object-oriented languages and generalized polyedra. Technical Report 92.7, Ecole des Mines de St-Etienne, June 1992.

[9] M. Beigbeder, D. Michelucci, and J.M. Moreau. Object-oriented languages and generalized polyedra. In *Third Eurographics Workshop on Object-Oriented Graphics*, pages 35–49, Champéry, Switzerland, October 1992.

[10] M.O. Benouamer. *Operations booléennes sur les polyèdres représentés par leurs frontières et imprécisions numériques*. PhD thesis, École des Mines de St Étienne, 1993.

[11] M.O. Benouamer, P. Jaillon, D. Michelucci, and J-M. Moreau. A lazy solution to imprecision in computational geometry. In *Proceedings of the 5th Canadian Conference on Computational Geometry*, pages 73–78, Waterloo, Canada, August 5-9, 1993.

[12] M.O. Benouamer, P. Jaillon, D. Michelucci, and J-M. Moreau. A lazy arithmetic library. In *Proceedings of the IEEE 11th Symposium on Computer Arithmetic*, Windsor, Ontario, June 30-July 2, 1993.

[13] M.O. Benouamer, P. Jaillon, D. Michelucci, and J-M. Moreau. Hashing lazy numbers. In *Proceedings of SCAN-93*, Vienna, Austria, September 1993.

[14] M.O. Benouamer, P. Jaillon, D. Michelucci, and J.M. Moreau. Hashing lazy numbers. *Computing*, 53(3-4):205–217, 1994.

[15] M.O. Benouamer, P. Jaillon, D. Michelucci, and J.M. Moreau. Hashing lazy numbers. Technical Report 93.6, Ecole des Mines de St-Etienne, April 1993.

[16] M.O. Benouamer, P. Jaillon, D. Michelucci, and J.M. Moreau. A lazy exact arithmetic. Technical Report 93.4, Ecole des Mines de St-Etienne, April 1993.

[17] M.O. Benouamer, P. Jaillon, D. Michelucci, and B. Péroche. Error-free boundary evaluation using lazy arithmetic: A detailed implantation. Technical Report 93.5, Ecole des Mines de St-Etienne, May 1993.

[18] M.O. Benouamer, D. Michelucci, and B. Péroche. Boundary evaluation using a lazy rational arithmetic. In *Proceedings of the 2nd ACM/IEEE Symposium on Solid Modeling and Applications*, pages 115–126, Montréal, Canada, 1993.

[19] M.O. Benouamer, D. Michelucci, and B. Péroche. Error-free boundary evaluation based on a lazy rational arithmetic: a detailed implementation. *Computer-Aided Design*, 26(6):403–416, June 1994.

[20] A. Borning. The programming language aspects of thinglab, a constraint oriented simulation laboratory. *ACM Transactions on Programming Languages and Systems*, 3:353–387, oct 1981.

[21] W. Bouma, I. Fudos, C. Hoffmann, J. Cai, and R. Paige. A geometric constraint solver. Technical Report CSD-TR-93-054, Department of Computer Science, Purdue University, August 1993.

[22] V. Bourgoin. *Synthèse en trois dmensions: Analyse et recherche de modéles de projection, inspirés des concepts développés au cours de l'histoire de l'Art.* PhD thesis, Université de Paris VIII, 1994.

[23] B. Bruderlin. Constructing three-dimensional geometric objects defined by constraints. In *Interactive 3D Graphics*, pages 111–129, October 1986.

[24] M. Buthion. Un programme qui résoud formellement des problèmes de constructions géométriques. *RAIRO Informatique*, 3(4):353–387, oct 1979.

[25] Berge C. *Graphs.* Bordas, Gauthier-Villars, 1983.

[26] Hoffman C., Hopcroft J., and Karasick M. Robust set operations on polyhedral solids. *IEEE Comp. Graphics and Appli.*, 9(6):50–59, Nov. 1989.

[27] J.C. Carrega. *Théorie des corps, la règle et le compas.* Hermann, 1989.

[28] S.C. Chou, W.F. Schelter, and J.G. Yang. Characteristic sets and grobner bases in geometry theorem proving. In *Workshop on Computer Aided Geometric Reasoning*, pages 29–56, INRIA, France, june 1987.

[29] T. Duff. Interval arithmetic and recursive subdivision for implicit functions and constructive solid geometry. *Computer Graphics (Proceedings Siggraph 1992)*, 26(2):131–138, July 1992.

[30] L.W. Ericson and C.K. Yap. The design of linetool a geometric editor. In *Symposium on Computational Geometry*, pages 83–92, 1988.

[31] S. Fortune and C. Van Wyk. Efficient exact arithmetic for computational geometry. In *Proceedings of the 9th ACM Symposium on Computational Geometry*, pages 163–172, San Diego, May 1993.

[32] M. Gangnet, J.C. Hervé, T. Pudet, and J.M. Van Thong. Incremental computation of planar maps. *ACM Computer Graphics (SIGGRAPH 89)*, 23(3):345–354, July 1989.

[33] M. Gangnet and D. Michelucci. Un outil graphique interactif. In *Proceedings of MICAD 84*, pages 95–110. Hermès, Feb.-Mar 1984.

[34] M. Gangnet and J.M. Van Thong. Robust boolean operations on 2d paths. In *Proceedings of COMPUGRAPHICS91*, volume 2, pages 434–443, Sesimbra, Portugal, 1991.

[35] D.H. Greene and F.F. Yao. Finite-resolution computational geometry. In *Proceedings of the 27th Annual Symposium on the Foundations of Computer Science*, pages 143–152, 1986.

[36] L. Guibas, D. Salesin, and J. Stolfi. Epsilon geometry: Building robust algorithms from imprecise computations. In *Proceedings of the 5th ACM Symposium on Computational Geometry*, pages 208–217, 1989.

[37] C. Hoffman, J. Hopcroft, and M. Karasick. Towards implementing robust geometric computations. In *Proceedings of the 4th ACM Symposium on Computational Geometry*, pages 106–118, 1988.

[38] J.F. Hughes, D.H. Laidlaw, and J.F. Trumbore. Constructive solid geometry for polyhedral objects. In *Proceedings of SIGGRAPH86*, volume 20, no 4, pages 161–180, ACM Computer Graphics, Aug. 1986.

[39] P. Jaillon. *Proposition d'une arithmétique rationnelle paresseuse et d'un outil d'aide à la saisie d'objets en synthèse d'images.* PhD thesis, École Nationale Supérieure des Mines de Saint-Étienne, 1993.

[40] P. Jaillon and J-M. Moreau. 'LEA', a lazy exact arithmetic: Implementation and related problems. Technical Report (*in preparation*), École Nationale Supérieure des Mines de Saint-Étienne, 1994.

[41] R. Jégou, D. Michelucci, and S. Ait Aoudia. Reduction of constraints systems. Technical Report 92.13, Ecole des Mines de St-Etienne, October 1992.

[42] M. Karasick, D. Lieber, and L. Nackman. Efficient Delaunay triangulation using rational arithmetic. Technical Report RC 14455, IBM, 1989.

[43] K. Kondo. Algebraic method for manipulation of dimensional relationships in geometric models. *Computer Aided Design*, 24(3):141–147, mars 1992.

[44] S. Kla Koué. *Calcul Parallèle et En-ligne des fonctions arithmètiques.* PhD thesis, École Normale Supérieure de Lyon et Université Claude Bernard Lyon 1, 1993.

[45] H. Lamure and D. Michelucci. Decomposition of constraints systems. *Soumis à publication.*

[46] H. Lamure and D. Michelucci. R'esolution de contraintes par une m'ethode homotopique. In *Actes des Journées de l'AFIG*, pages 95–104, 1994.

[47] H. Lamure and D. Michelucci. Solving constraints systems by homotopy. In *Proceedings of 3^o ACM/IEEE Symposium on Solid Modeling and Applications*, May 1995.

[48] D. Lazard. Systems of algebraic equations: algorithms and complexity. Technical Report LITP 92.20, LITP, université Paris VI, VII, CNRS, mars 1992.

[49] Z. Li and V.J. Milenkovic. Constructing strongly convex hulls using exact or rounded arithmetic. In *ACM-SIAM Symposium on Discrete Algorithms*, 1989.

[50] R. Light, V. Lin, and D.C. Gossard. Variational geometry in cad. *Computer Graphics*, 15(3):171–177, aug 1981.

[51] L. Lovasz and M.D. Plummer. *Matching Theory.* North-Holland, 1986.

[52] D. Michelucci. An ϵ-arithmetic for removing degeneracies. *Soumis à publication.*

[53] D. Michelucci. *Les représentations par les frontières: quelques constructions; difficultés rencontrées.* PhD thesis, École Nationale Supérieure des Mines de Saint-Étienne, 1987.

[54] D. Michelucci and J.M. Moreau. Lazy arithmetic. *Soumis à publication.*

[55] D. Michelucci and B. Péroche. Représentation interactive d'arbres csg. In *Proceedings of MICAD 87*, pages 535–548. Hermès, 1987.

[56] D. Michelucci and B. Péroche. Visualisation interactive d'arbres csg. Technical Report 86.09, Ecole des Mines de St-Etienne, Septembre 1986.

[57] V.J. Milenkovic. Double precision geometry: A general technique for calculating line and segment intersections using rounded arithmetic. In *Proceedings of the 30th Annual Symposium on the Foundations of Computer Science*, 1989.

[58] J-M. Moreau. *Facétisation et hiérarchisation de la représentation par segments d'un graphe planaire dans le cadre d'une arithmétique mixte.* PhD thesis, École Nationale Supérieure des Mines de Saint-Étienne, 1990.

[59] K. Murota. *Systems Analysis by Graphs and Matroids, Structural Solvability and Controllability.* Springer-Verlag, 1987.

[60] A. Nahed. *Modèlisation $2D\frac{1}{2}$ hiérarchique basée sur les cartes planaires; réalisation et évaluation d'une interface graphique utilisant cette modélisation.* PhD thesis, Ecole des Mines, St Etienne, France, 1994.

[61] J. Nakagawa, H. Sato, K. Toshimitsu, and F. Yamagushi. An adaptive error-free computation based on the 4x4 determinant. *The Visual Computer*, 9:173–181, 1993.

[62] G. Nelson. Juno, a constraint-based graphic system. In *ACM Siggraph Conference Proceeding*, 1985.

[63] J.C. Owen. Algebraic solution for geometry from dimensional constraints. In *Proceedings of the Symposium on Solid Modeling Foundations and CAD/CAM Applications*, pages 397–407, 1991.

[64] B. Péroche, J. Argence, D. Ghazanfarpour, and D. Michelucci. *La synthèse d'images.* Hermès, 1988.

[65] A. Recski. *Matroid theory and its applications.* Springer Verlag, 1986.

[66] J.M. Snyder. Interval analysis for computer graphics. *Computer Graphics (Proceedings Siggraph 1992)*, 26(2):121–130, July 992.

[67] K. Sugihara and M. Iri. A solid modeling system free from topological inconsistencies. Technical Report RM 89-03, University of Tokyo, 1989.

[68] I.E. Sutherland. Sketchpad, a man machine graphical communication system. In *AFIPS, Spring Joint Computer Conference*, pages 329–346, Detroit, Michigan, may 1963.

[69] A. Verroust, F. Schonek, and D. Roller. Rule oriented method for parametrized computer aided design. *Computer Aided Design*, 24(3):531–540, Oct 1992.

[70] F. Winkler. Algorithms in polynomial ideal theory and geometry. In *2nd Austrian-Hungarian Informatics Conf.*, Retzhof Austria, sep 1988.