Résolution de contraintes géométriques par homotopie *

Dominique Michelucci, Hervé Lamure

Résumé

La modélisation géométrique par contraintes conduit à des systèmes d'équations qui sont généralement résolus par l'algorithme de Newton-Raphson (ou une de ses variantes), à partir d'une situation fournie interactivement par l'utilisateur. Mais, trop souvent, cette méthode ne converge pas, ou converge vers une solution non voulue après un parcours chaotique. L'application d'une méthode homotopique dans de tels cas donne des résultats bien plus satisfaisants.

1 Introduction

La modélisation par contraintes est utilisée depuis longtemps en CAO pour simplifier le travail des concepteurs. Elle permet de ne plus avoir à spécifier des positions précises pour les objets manipulés, mais seulement des relations entre ces objets. On ne parlera ici que de contraintes géométriques (*ie* des relations telles que distances, tangences, angles, incidences,...) entre figures (typiquement points, segments, coniques, splines, courbes de Bézier,...en 2 dimensions auxquelles on peut rajouter les plans, tores, quadriques, carreaux de Bézier,...en 3D). Cette modélisation conduit à de grands systèmes d'équations, généralement algébriques. Le problème est de pouvoir les résoudre efficacement.

L'idée de contraintes existait déjà dans les travaux de Sutherland [Sut63] et Borning [Bor81]; depuis de nombreuses voies ont été explorées pour résoudre des problèmes de plus en plus difficiles. On peut classer grossièrement les méthodes de résolution de contraintes en trois grandes catégories (non exclusives): méthodes de décomposition, résolution symbolique, et algorithmes numériques.

Les méthodes de décomposition réduisent les systèmes de contraintes en d'autres plus simples, puis les solutions des sous-systèmes sont fusionnées. 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 les systèmes solubles à la règle et au compas -comme le problème d'Appolonius-. Dans ces cas une formule explicite permet de résoudre; cependant des sous-systèmes irréductibles plus complexes devront 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 [But79] ou Verroust *et al* [VSR92], soit par un mécanisme d'unification fourni par un langage de programmation tel que Prolog, comme pour Brüderlin [Bru86], soit enfin par une méthode de graphes, comme pour, entre autres, Owen [Owe91], Bouma *et al* [BFH+93] , ou Ait *et al* [AAJM93].

Les méthodes symboliques sont utilisées, par exemple, par Ericson et Yap [EY88] ou par Kondo [Kon92]. Ces méthodes utilisent typiquement les bases de Gröbner, ou un genre de résultant, ou encore une technique d'élimination. Si ces méthodes sont exactes elles sont par

*Ce rapport est la version française d'un article présenté au third symposium on solid modeling.

contre particulièrement lentes: d'après Lazard [Laz92] il est définitivement impossible de calculer les bases de Gröbner d'un système polynomial irréductible de degré 2 en 10 inconnues et variables. Ces méthodes semblent plus appropriées à la démonstration automatique de théorèmes, comme par exemple dans [CSY87, Win88].

Les méthodes numériques restent 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 [Bor81], Nelson [Nel85], ou Light *et al* [LLG81]. 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 [AAJM93] pour un exemple) qui accélèrent la résolution.

Cependant il reste un problème bien connu: si la méthode de Newton-Raphson fonctionne généralement bien, parfois -trop souvent!-, elle ne converge pas, ou converge vers une solution non voulue après un parcours chaotique. L'utilisateur ne sait alors pas vraiment quoi faire, si ce n'est modifier légèrement sa figure initiale, jusqu'à ce que la méthode marche, ou si elle marche...

Ce papier entend montrer qu'une méthode homotopique est bien plus satisfaisante dans de tels cas. Son comportement est facile à prédire, intuitif, et s'explique de lui même.

La figure 1 montre un exemple typique d'échec de la méthode de Newton-Raphson: on voudrait que les six cercles, tout en restant tangents entre eux, le soient aussi aux parois du triangle. Comme sur la figure 2 qui montre quelques étapes affichées lors de la résolution du même problème par homotopie.

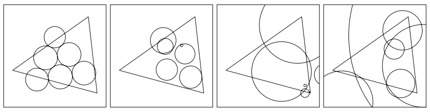

FIG. 1 - *Echec de la méthode de Newton*

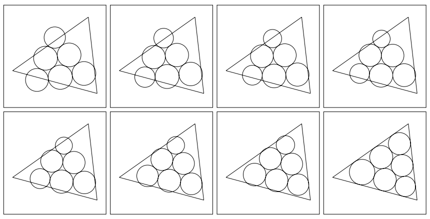

FIG. 2 - *L'homotopie a un comportement naturel*

Un autre argument en faveur de nos dires repose sur le système d'équations

$$\left\{ \begin{array}{lcl} x(x^2-3y^2) & = & 1 \\ y(3x^2-y^2) & = & 0 \end{array} \right. \quad x, y \in \mathbb{R}$$

Cette équation n'est en fait que l'équivalent, utilisant des variables réelles, de l'équation complexe $z^3 - 1 = 0$. La figure 3 montre les bassins d'attraction pour la méthode de Newton-Raphson et l'homotopie. Pour Newton-Raphson les bassins sont fractals [PR86], ce qui explique pourquoi il est si difficile à l'utilisateur de prédire vers quelle solution cette méthode va converger. A l'inverse l'homotopie converge vers la solution la plus proche[1]. Les bassins de l'homotopie ont des frontières lisses: se sont des ensembles semi-algébriques[2] quand le système à résoudre est algébrique. Plus généralement l'homotopie converge plus souvent que Newton-Raphson vers la solution intuitivement la plus proche de

[1]Sur cet exemple les bassins d'attraction pour l'homotopie sont les cellules de Voronoï des points solutions, ceci n'est pas vrai dans le cas général.

[2]Un ensemble semi-algébrique est la projection d'un ensemble algébrique.

l'approximation initiale, ce qu'il est bien sûr impossible de prouver rigoureusement vu que "intuitivement proche" n'a pas de définition mathématique.

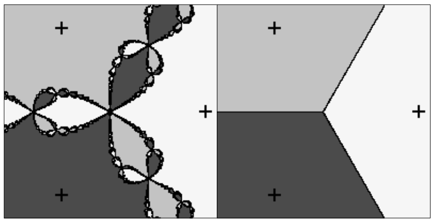

FIG. 3 - $z^3-1=0$, $z \in \mathbb{C}$: *Bassins d'attraction pour les méthodes de Newton et homotopique*

Cet article donne d'abord un exposé simplifié sur l'homotopie (aussi appelée méthode de continuation). Puis il commente nos expériences sur un éditeur géométrique par contraintes, avec les questions liées et les extensions possibles. Notez que nous nous sommes restreint à un éditeur $2D$ pour des raisons de simplicité, mais l'homotopie, tout comme la méthode de Newton-Raphson, s'applique indifféremment à toutes les dimensions.

La résolution par homotopie est maintenant une méthode classique de l'analyse numérique, elle était déjà utilisée par Lahaye en 1934, et peut être même avant (voir le récent tour d'horizon de Allgower et Georg [AG93] pour un historique détaillé). Dans le domaine de l'informatique graphique, Dobkin *et al* [DLTW90] suivent des courbes dans $\mathbb{R}^n$ grâce à une méthode inspirée par un cas spécial d'homotopie: l'interpolation linéaire par morceaux. En CAO, elle a été utilisée, pour des applications de robotique et de cinématique, par Morgan, Wampler et Sommese [Mor83, WMS90], mais eux recherchent toutes les solutions complexes des systèmes polynomiaux survenant dans ces domaines. En modélisation géométrique utilisant des surfaces à formes libres, un genre spécial d'homotopie sert à suivre les courbes d'intersection de surfaces dans $\mathbb{R}^3$: la *linéarisation par morceaux* (*marching method*) [Pat92]. Curieusement, en modélisation par contraintes, il semble que jusqu'à maintenant personne n'utilise de méthode homotopique, sauf une continuation restreinte présentée au chapitre 2.3.4.

2 Généralités sur l'homotopie

On pourra trouver un exposé plus complet dans [AG93].

Soit $G(X) = 0$ un système de n equations indépendantes à n inconnues, on a $X = (x_1, x_2, \ldots, x_n)$ et $G = (g_1, g_2, \ldots, g_n)$. Un ensemble de contraintes géométriques bien-contraint fournit un tel système.

Supposons maintenant que la solution $S = (s_1, s_2, \ldots, s_n)$ d'un autre système $F(X) = 0$ avec $F = (f_1, f_2, \ldots, f_n)$ soit connue, et que F soit, dans un certain sens, "proche" de G. Dans notre application S n'est autre que le vecteur des valeurs (coordonnées de points, rayon de cercle,...) de l'approximation initiale donnée par l'utilisateur. On définit alors le système F par $F(X) = G(X) - G(S)$, par construction S est bien solution de $F(X) = 0$. F et G sont alors plongés dans une *homotopie*:

$$H(t, X) = tG(X) + (1-t)F(X)$$

telle que $H(0, X) = F(X)$ et $H(1, X) = G(X)$. H est une interpolation linéaire entre F et G. Certaines homotopies utilisent une interpolation non linéaire, mais sont en dehors du champ de cet article (voir [AG93] pour plus de détails).

Remarque: Dans notre cas, du fait que $F(X) = G(X) - G(S)$, il s'ensuit que $H(t, X) = G(X) - (1-t)G(S)$. Le terme $(1-t)G(S)$ est une fonction de t seul, pas de X, cependant la théorie de l'homotopie a été développée dans le cas général pour tout F, et la suite de cette présentation n'utilise pas cette particularité.

Le système H possède $n+1$ inconnues et n équations. Si $P = (t_p, X_p)$ est tel que $H(P) = 0$, et si P est un point régulier de $H = 0$ (*i.e.* le jacobien $H'(P)$ est de rang maximal), alors, d'après le théorème de la fonction implicite, $H^{-1}(0)$ est localement paramétrable par t au point P. En termes plus géométriques $H(t, X) = 0$

définit une courbe dans un espace de dimension $n+1$, passant par P et localement paramètrée par t. Un tel point est connu: il s'agit de $P=(0,S)$. L'idée principale de la résolution par homotopie est de suivre la *courbe homotopique* (on dit aussi *chemin homotopique*), démarrant en $(t=0, X=S)$. Si le chemin homotopique passe par un point (t,X) où $t=1$, alors une solution de $H(1,X)=0$, et donc de $G(X)=0$, est trouvée. Les méthodes pour suivre les courbes d'homotopie sont énumérées plus loin. La figure 4 montre un échantillonnage de 12^2 courbes homotopiques $3D$ correspondant à l'exemple de la figure 3, le damier est dans le plan $t=0$.

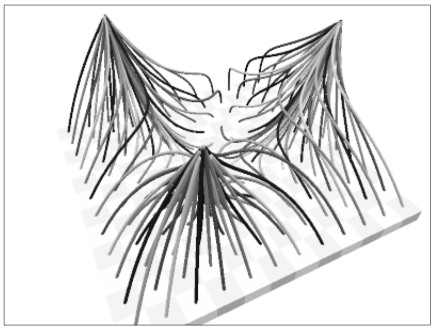

FIG. 4 - *$z^3-1=0$, $z \in \mathbb{C}$: Courbes homotopiques en $3D$.*

2.1 Considérations topologiques

Si une courbe homotopique ne contient que des points réguliers, sa topologie peut être celle d'un cercle (*ie* la courbe est une boucle), ou celle d'une droite (*ie* la courbe vient de l'infini et repart à l'infini). Bien sûr, dans chaque cas, elle peut croiser ou non l'hyperplan d'équation $t=1$ (*cf* figure 5). Une courbe homotopique peut aussi contenir des points irréguliers: le jacobien ne sera pas alors de rang maximal. Ces points sont appelés *points de bifurcation*: les plus simples sont quadratiques, tels que deux courbes de $\mathbb{C}^{n+1}$ se croisent en un point $Q=(t_0,X_0)\in\mathbb{C}^{n+1}$ (Q est une solution régulière de: $\{H(Q)=0$ et $det(H_X{}'(Q))=0\}$). Les deux vecteurs tangents Φ et $i\Phi$ des deux branches sont orthogonaux au point Q. Les *coudes* (*turning points*) sont des cas spéciaux de points de bifurcation quadratiques que l'on rencontre typiquement lorsque F et G sont des systèmes réels [LW93]: une courbe est réelle, l'autre est imaginaire ou complexe, la figure 6 en montre deux exemples. Le vecteur tangent en ces points est horizontal, c'est à dire que sa composante en t est nulle.

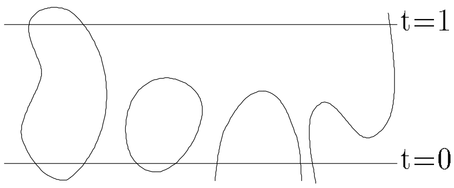

FIG. 5 - *Quelques exemples de courbes homotopiques.*

2.2 Méthode homotopique

Dans notre cas, un point de départ pour l'homotopie est connu, et nous sommes seulement intéressés par la courbe homotopique passant par ce point. Cependant, dans le cas général on veut trouver toutes les solutions (même complexes) d'un système donné, et sans point de départ. On procède alors en deux étapes. On commence par construire un premier système d'équations facilement soluble, et possédant au moins autant de solutions que le système à résoudre, qui nous donne les points de départ pour l'homotopie. Ce nombre de solutions peut être borné classiquement par le produit des degrés d'après le théorème de Bezout, ou, plus précisément, par le polytope de Newton et les bornes BKK [VVC93]. La seconde étape consiste à suivre les chemins depuis les points de départ.

2.3 Méthodes de suivi de courbe

2.3.1 Homotopie complexe montante

Pour les systèmes de contraintes de la CAO seules les racines réelles de G sont intéressantes, et les chemins suivis sont des courbes dans $\mathbb{R}^{n+1}$. Dans d'autres domaines, les racines complexes Z de G sont demandées, les chemins homotopiques sont alors des courbes dans $\mathbb{R} \times \mathbb{C}^n$ (puisque, traditionnellement, t va de 0 à 1 en restant dans $\mathbb{R}$). Dans ce dernier cas, il est possible de "monter" le long des courbes d'homotopie, en partant de $(t = 0, Z = Z(0))$, et en suivant $Z(t)$ lorsque t croît de 0 à 1. Par linéarisation de H en un point connu (t, Z) de la courbe on obtient:

$$H(t+\Delta t, Z+\Delta Z) \approx H(t, Z)+H_t{}'\Delta t+H_Z{}'\Delta Z$$

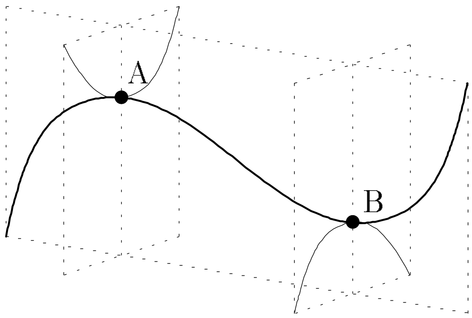

FIG. 6 - *Une courbe homotopique réelle, en ligne grasse, ayant deux coudes A et B: le vecteur tangent a une composante en t nulle. En ligne mince, les branches complexes correspondantes (ici imaginaires pures). Les équations pourraient être:* $F(X) = 3X^3 - X + \frac{1}{2}$, $G(X) = 3X^3 - X - \frac{1}{2}$, $H(t, X) = 3X^3 - X + \frac{1}{2} - t$. $A_X = \frac{13}{18}$, $A_t = -\frac{1}{3}$, $B_X = \frac{5}{18}$, $B_t = \frac{1}{3}$.

On utilise la prédiction[3]:

$$\Delta Z = -(H_Z{}')^{-1} H_t{}'\Delta t$$

puis, laissant t constant (donc $\Delta t = 0$), on corrige Z en itérant $\Delta Z := -(H_Z{}')^{-1} H(t, Z)$, $Z := Z + \Delta Z$, jusqu'à ce que $|\Delta Z|$ soit suffisament proche de 0.

Cette méthode pose problème lorsqu'un point de bifurcation est rencontré, une solution possible est d'utiliser alors l'homotopie perturbée:

$$H(t, Z) = t\gamma G(Z) + (1 - t)F(Z)$$

où $\gamma = \rho e^{i\theta}$ est un nombre complexe aléatoire: il est prouvé qu'il existe seulement un nombre fini de θ pour lesquels H possède des points de bifurcation quand $t \in [0, 1[$. Donc, avec probabilité 1, l'homotopie perturbée élimine les points de bifurcation. Bien sûr cette perturbation ne peut enlever les éventuels points singuliers de $G = 0$, et les points de bifurcation correspondants quand $t = 1$.

2.3.2 Prédiction-correction

Pour suivre le chemin homotopique depuis un point donné M_k la méthode

[3] Δt peut être adapté pour contrôler la taille du pas $|(\Delta t, \Delta Z)|$.

dite de prédiction-correction commence par calculer T_k, le vecteur tangent (ou au moins une approximation) de la courbe en M_k, puis "prédit" que le point $P = M_k + \epsilon T_k/|T_k|$ est proche de la courbe, et corrige P par Newton-Raphson (ou une de ses variantes comme la méthode de la sécante, ou encore en utilisant la pseudo-inverse de Moore-Penrose) ou une méthode de gradient pour obtenir le point M_{k+1}, le point de la courbe le plus proche de P. On ne détaillera pas plus ici cette méthode qui est très similaire à la méthode utilisée en CAO pour suivre les courbes d'intersection entre surfaces en modélisation géométrique $3D$ [Pat92, Hof90].

Une difficulté pratique (et un problème théorique d'analyse numérique difficile) est le choix de l'ϵ: s'il est trop grand l'étape "correction" peut échouer, et s'il est trop petit le suivi devient très lent (de plus des problèmes d'imprécision numérique peuvent apparaître). De nombreuses recherches ont donc été entreprises pour choisir automatiquement un ϵ correct [AG93, Yak95]. Mais, vu nos besoins limités, nous avons trouvé l'heuristique suivante suffisante et facile à implanter: à chaque pas la pseudo-inverse de H' est actualisée, et on choisi $\epsilon = 0,05$; si l'étape de correction ne converge pas assez vite (disons en moins de 4 itérations), on divise ϵ par 2 et on recommence. Soit d la distance entre le point de départ M_k et le point prédit P_0, soient P_1, P_2, P_3 et P_4 les corrections successives de P_0, si la distance $|P_0P_i|$ est supérieure à d, on considère que l'on diverge. Il y a convergence quand $|P_iP_{i-1}|$ est inférieure à $0,02d$, et que l'angle entre les vecteurs tangents aux points M_k et P_i est inférieur à 10 degrés.

Notre modeleur expérimental utilise la méthode de prédiction-correction principalement parce que nous avions aussi besoin de Newton-Raphson pour pouvoir comparer son comportement avec celui de l'homotopie. Mais cette méthode nécessite le calcul[4] de l'inverse du jacobien H'. Les autres méthodes ne requièrent que l'évaluation de H en certains points.

2.3.3 Interpolation linéaire par morceaux

Une autre méthode pour suivre des chemins homotopiques est connue sous le nom d'interpolation linéaire par morceaux. En informatique graphique, une variante de cette méthode a été utilisée par Dobkin *et al* [DLTW90] pour tracer des courbes dans $\mathbb{R}^n$.

Dans $\mathbb{R}^2$ cette méthode est très simple. Pour tracer la courbe homotopique $H(t,x) = 0$, $\mathbb{R}^2$ est découpé en triangles de coté ϵ. Supposons qu'un premier triangle ABC traversé par H soit connu: H entre par le coté AB et ressort par AC si $H(A) > 0$, $H(B) < 0$, et $H(C) < 0$, ou l'opposé. Alors le prochain triangle coupé par H est ACB', où $B' = A - B + C$ est le symétrique de B par rapport à AC. Dans ABC, H est approché par l'unique application linéaire $L(t,x)$ de $\mathbb{R}^2$ dans $\mathbb{R}$ telle que $L(A) = H(A)$, $L(B) = H(B)$ et $L(C) = H(C)$. La courbe dans ABC est approchée par le segment $\{L(x,y) = 0\} \cap ABC$.

Dans $\mathbb{R}^{n+1}$ l'espace est decoupé en "hyper tétraèdres" (ou simplexes). Supposons que l'on connaisse un premier simplexe T traversé par la courbe homotopique, supposons aussi que les valeurs des n fonctions définissant H soient disponibles aux sommets de T, elles définissent une application linéaire L de $\mathbb{R}^{n+1}$ dans $\mathbb{R}^n$. Dans T, on approche H par L et la courbe homotopique par le segment $\{L(t, x_1, x_2, \ldots, x_n) = 0\} \cap T$. On en déduit l'hyperfacette de T par laquelle cette courbe le quitte, et on suit le chemin homotopique dans le simplexe voisin correspondant.

2.3.4 Une méthode homotopique restreinte

Parfois,
en modélisation par contraintes, une méthode restreinte de suivi homotopique est utilisée [Ver90]: supposons qu'une première solution X_0 d'un système de contraintes c_0 soit connue, mais que la valeur d'un paramètre $p \in \mathbb{R}$ (une lon-

[4] La définition symbolique du jacobien H' est calculée pour chaque système de contraintes.

gueur, un angle, un rayon,...) doive être changée de, disons, p_0 en p_1. Il est pratique de voir le système de contraintes c comme une fonction de p: $c = C(P)$ et donc $c_0 = C(p_0)$, et de résoudre alors $c_1 = C(p_1)$. Lorsqu'on essaie de résoudre directement $C(p_1)$ par Newton-Raphson avec X_0 comme valeur initiale, il existe un risque de divergence. Une idée naturelle est de commencer par résoudre, par exemple, $C(0,9p_0 + 0,1p_1)$ pour obtenir $X_{0,1}$ en utilisant une des variantes de la méthode de Newton-Raphson, puis, partant de $X_{0,1}$, résoudre $C(0,8p_0 + 0,2p_1)$ pour obtenir $X_{0,2}$,... Jusqu'à obtenir la solution X_1 de $C(p_1) = 0$. On peut bien sûr généraliser à un nombre quelconque de pas, ainsi qu'un nombre quelconque de paramètres. Il s'agit en fait d'une variante naïve de l'homotopie montante.

Cette méthode est fortement limitée: elle fonctionne seulement lorsque la courbe à suivre n'a pas de coude, et on peut trouver des coudes même dans des exemples très simples. L'homotopie montante n'est pertinente que dans le cadre des nombres complexes, en utilisant en plus une perturbation pour éliminer les points de bifurcation.

2.4 Bassins d'attraction

Un bassin d'attraction pour l'homotopie est un ensemble connexe de points $S \in \mathbb{R}^n$ donnant la même solution ($t = 1, X_1$), ou pas de solution, quand l'homotopie est appliquée avec comme point de départ ($t = 0, S$). Deux bassins voisins sont séparés par des branches menant à des points de bifurcation. Ces derniers sont solutions du système algébrique en $n + 1$ equations et inconnues $H(t, X) = det(H_X'(t, X)) = 0$ qui est un ensemble algébrique (si G est algébrique). La projection sur l'hyperplan d'équation $t = 0$ donne alors un ensemble semi-algébrique. Le lecteur peut aisément vérifier que sur l'exemple de la figure 3 les frontières entre les bassins sont constituées par des demi-droites d'équation $y = 0$, et $y = \pm x\sqrt{3}$.

Il est bien connu que les bassins d'attraction de Newton-Raphson sont fractals (voir figure 3 par exemple), c'est ce qui explique le comportement chaotique de cette méthode. Au contraire les bassins plus lisses de l'homotopie impliquent un comportement plus sympathique.

2.5 Homotopie sous-contrainte

Pour qu'un modeleur sous contraintes soit réellement convivial et interactif il doit aussi pouvoir résoudre les systèmes sous-contraints, et pas seulement les bien-contraints. Mais s'il y a n inconnues (sans t) et u équations manquantes, alors $H(t, X) = 0$ ne décrit plus une courbe dans $\mathbb{R}^{n+1}$ mais une hypersurface de dimension $u + 1$. La procédure est maintenant la suivante: au point initial $M_0 = (0, S)$ on projette le vecteur $V = (t = 1, x_1 = x_2 = \ldots = x_n = 0)$ sur l'espace tangent à cette hypersurface pour trouver T_0; le point prédit est alors $M_0 + \epsilon T_0/|T_0|$ et une étape de correction nous donne M_1. Au point $M_k, (k > 0)$ on projette le précédent vecteur tangent T_{k-1} sur l'espace tangent de $H(M_k) = 0$ pour obtenir le vecteur tangent T_k et on prédit le point $M_k + \epsilon T_k$. On suit en fait (aux erreurs d'échantillonnage et d'imprécision numérique près) la courbe géodésique ayant un vecteur tangent montant T_0 au point $(0, S)$. L'homotopie ainsi définie possède un comportement intuitif: supposons, par exemple, qu'un point P doive appartenir à une droite donnée L, mais que le point donné initialement P_0 n'y appartienne pas: l'utilisateur voit alors le point s'approcher de la droite en suivant le plus court chemin.

Une autre possibilité, que nous n'avons pas essayée, est de projeter le vecteur montant V sur l'espace tangent à $H(M_k) = 0$ à chaque pas. Il y a cependant un problème lorsque $det(H_X'(M_k)) = 0$, c'est à dire lorsque l'on se trouve sur un apex local de H.

2.6 Solutions à l'infini

Les solutions à l'infini (par exemple le système $xy = 1$, $y = 0$ a comme solution $x = \pm\infty$) posent un problème lors de l'application de méthodes homotopiques, puisque les chemins vers ces solutions sont de longueur infinie. Nous n'avons jamais rencontré ce problème lors de nos expérimentations, cependant il est facilement résolu par l'homogénéisation du système. Supposons par exemple que les inconnues soient les coordonnées (x, y) d'un certain point. Il suffit d'utiliser les coordonnées homogènes (X, Y, H) pour ce point, en posant: $xH = X$, $yH = Y$. On transforme ensuite les contraintes $g_i(x, y) = 0$ en $G_i(X, Y, H) = H^{deg(g_i)} g_i(X/H, Y/H) = 0$. On ajoute une nouvelle contrainte, telle que: $X^2 + Y^2 + H^2 = 1$. Et enfin on résout: toutes les solutions du système homogène sont finies. Sur l'exemple précédent, le système homogénéisé est: $XY = H^2$, $Y = 0$, $X^2 + Y^2 + H^2 = 1$, on trouve comme solution: $X = \pm 1$, $Y = H = 0$. Pour plus de détails voir [AG93, Mor86].

3 Un modeleur par contraintes $2D$

3.1 Description

Nous avons implanté, en Lelisp, un modeleur $2D$ expérimental pour comparer les comportements respectifs de l'homotopie et de la méthode de Newton-Raphson. L'utilisateur crée des points, segments, cercles,... de manière interactive comme dans MacDraw ou Xfig. Il peut ensuite spécifier des contraintes. Certaines sont prédéfinies: distance ou angle entre éléments, relations de tangence ou d'incidence. Mais on peut aussi spécifier ses propres contraintes en donnant explicitement une formule algébrique. Les coordonnées peuvent être déclarées comme modifiables ou non: le modeleur ne peut changer que les coordonnées modifiables, qui sont les inconnues dont la valeur va être utilisée au départ de la résolution. L'utilisateur peut appeler la résolution du système par Newton-Raphson, ou par homotopie. Ces deux méthodes acceptent que les systèmes à résoudre soient bien- ou sous-contraints: l'utilisateur peut donc ajouter incrémentalement des contraintes et résoudre. Pour l'instant le modeleur considère les systèmes sur-contraints comme des erreurs de l'utilisateur, et lui donne un avertissement.

Lorsque la résolution par Newton-Raphson échoue, ou donne une solution non voulue, l'utilisateur récupère l'état précédent et utilise l'homotopie. Généralement le fonctionnement est bien meilleur. Parfois pourtant l'homotopie peut converger vers une solution que l'on ne désire pas, mais le comportement très compréhensible de l'homotopie, que l'utilisateur peut observer puisque les étapes intermédiaires de la résolution sont affichées, permet de se rendre facilement compte du problème et de le corriger. Par exemple si un cercle passe du mauvais coté d'une droite durant la résolution, l'utilisateur interrompt le processus (ou attend sa fin), rétablit l'état précédent, déplace le cercle ou la droite, et enfin appelle à nouveau la résolution: ça marche!

Dans nos premières expériences l'homotopie fonctionnait pratiquement toujours du premier coup, puis les ratés se sont faits de plus en plus fréquents. La raison en est que, voyant l'homotopie fonctionner si bien, nous devenions de plus en plus paresseux, et donnions des "approximations" initiales de plus en plus eloignées de la solution désirée.

Nous n'avons jamais rencontré de points de bifurcation, sauf lorsque nous le faisons à dessein pour tester le logiciel. Typiquement ces situations se produisent lorsqu'il y a plusieurs solutions symétriques, et que la donnée de départ comporte la même symétrie. Par exemple si un point P possède deux solutions symétriques P_1 et P_2 relativement à une droite L, et si le point P initial se trouve sur L, alors l'homotopie ne peut "choisir" entre deux chemins symétriques, un menant à P_1 et l'autre à P_2: la donnée initiale est elle même un point de bifurcation. Une légère perturbation au départ est suffisante pour briser la symétrie et supprimer le point de bifurcation.

3.2 Problèmes ouverts

Contrôle: Quand un système n'a pas de solution, cela se voit facilement (en supposant que l'on soit en coordonnées homogènes pour tenir compte des solutions à l'infini): l'homotopie boucle. Bien que probablement faisable, la détection automatique des bouclages n'est pas encore implantée, l'utilisateur doit interrompre de lui même le processus infini (son travail n'est pas perdu!).

Diagnostique et décomposition: Notre modeleur fournit déjà certains outils pour diagnostiquer les systèmes de contraintes: reconnaissance des inconnues sous-, sur- et bien-contraintes [AAJM93] et de la rigidité (comme dans [Hen92] par exemple). Ces outils sont essentiels pour aider l'utilisateur à "déverminer" un système de contraintes complexe. De plus ils permettent de décomposer les grands systèmes de contraintes en plusieurs petits, et donc d'accélérer la résolution. L'homotopie, tout comme la méthode de Newton-Raphson, est compatible avec de telles méthodes de décomposition. Cependant, ces dernières ne sont pas encore disponibles dans l'implantation actuelle.

Vitesse: Notre implantation de l'homotopie n'est pas aussi rapide qu'elle pourrait l'être (même si la résolution des contraintes reste bien plus rapide que leur spécification interactive), surtout avec plus d'une cinquantaine d'inconnues. Notre but premier était de vérifier la pertinence de l'homotopie, pas sa vitesse. La lenteur de notre implantation n'est donc pas significative. Plus intéressant est le nombre d'étapes de prédiction-correction effectuées en moyenne: le plus souvent une vingtaine suffisent, mais 60 ou 70 peuvent être nécessaires lorsque le chemin suivi passe à proximité d'un autre, *ie* à des quasi points de bifurcation (voir figure 4). En moyenne, dans la pratique, l'homotopie est environ 20 fois plus lente que la méthode de la sécante utilisée à chaque étape.

Imprécision: En regardant attentivement les bassins d'attractions de la figure 3 on peut remarquer quelques petites erreurs le long des frontières. Idéalement les frontières entre bassins devraient être des demi-droites sur cet exemple. Ce problème est surtout dû à des phénomènes d'imprécision numérique. Dans certains domaines ce genre de problèmes est un inconvénient sérieux à l'utilisation de l'homotopie, car la confusion entre chemins très proches (mais distincts) peut conduire à des incohérences logiques ou topologiques. Cependant, pour notre application, et dans une utilisation interactive, ce n'est pas un problème grave.

Equation non algébriques: L'homotopie n'est pour le moment utilisée qu'avec des contraintes algébriques. Elle peut de la même manière fonctionner avec des équations transcendantes.

Inéquations: L'homotopie ne permet normallement de résoudre que les systèmes à base d'équations, pas d'inéquations. Il reste cependant toujours la possibilité de transformer les inéquations en équations en appliquant le schéma suivant : supposons que l'on veuille résoudre $f(X) \geq 0$, ceci peut aussi s'écrire $f(X) - a^2 = 0$ où a est une inconnue réelle auxiliaire. Nous savons déjà comment résoudre les systèmes sous-contraints induits par ce genre de manipulation. Cette astuce fonctionne bien, cependant, à cause de l'imprécision inhérente aux méthodes numériques, il n'est pas possible de faire la distinction entre $f(X) \geq 0$ et $f(X) > 0$.

4 Conclusion

Nous sommes convaincus que l'homotopie sera bientôt populaire parmi les modeleurs géométriques par contraintes, et même parmi les descendants de MacDraw ou Xfig. Elle n'est pas réellement difficile à implanter, son comportement est plus prévisible, plus intuitif que celui de Newton-Raphson. Enfin elle est compatible avec une utilisation interactive et les méthodes de décomposition. Comme exemple nous avons implanté un modeleur expérimental $2D$, mais l'homotopie (comme la méthode de Newton-Raphson) s'applique aussi à la modélisation $3D$.

Références

[AAJM93] S. Ait-Aoudia, R. Jegou, and D. Michelucci. Reduction of constraint systems. In *Compugraphic*, pages 83–92, Alvor, Portugal, 1993.

[AG93] E.L. Allgower and K. Georg. Continuation and path following. *Acta Numerica*, pages 1–64, 1993.

[BFH+93] 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.

[Bor81] 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.

[Bru86] B. Bruderlin. Constructing three-dimensional geometric objects defined by constraints. In *Interactive 3D Graphics*, pages 111–129, October 1986.

[But79] M. Buthion. Un programme qui résoud formellement des problèmes de constructions géométriques. *RAIRO Informatique*, 3(4):353–387, oct 1979.

[CSY87] 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.

[DLTW90] D.P. Dobkin, S.V.F. Levy, W.P. Thurston, and A. Wilks. Contour tracing by piecewise linear approximations. *ACM Transaction on Graphics*, 9(4):389–423, oct 1990.

[EY88] L.W. Ericson and C.K. Yap. The design of linetool a geometric editor. In *Symposium on Computational Geometry*, pages 83–92, 1988.

[Hen92] B. Hendrickson. Conditions for unique realizations. *SIAM J. Computing*, 21(1):65–84, feb 1992.

[Hof90] C.M. Hoffmann. A dimensionality paradigm for surface interrogations. *Computer Aided Geometric Design*, 7:517–532, 1990.

[Kon92] K. Kondo. Algebraic method for manipulation of dimensional relationships in geometric models. *Computer Aided Design*, 24(3):141–147, mars 1992.

[Laz92] D. Lazard. Systems of algebraic equations: algorithms and complexity. Technical Report LITP 92.20, LITP, université Paris VI, VII, CNRS, mars 1992.

[LLG81] R. Light, V. Lin, and D.C. Gossard. Variational geometry in cad. *Computer Graphics*, 15(3):171–177, aug 1981.

[LW93] T.Y Li and Xiaoshen Wang. Solving real polynomial systems with real homotopies. *Mathematics of Computation*, 60(202):669–680, 1993.

[Mor83] A.P. Morgan. *Solving Polynomial Systems Using Continuation for Scientific and Engineering Problems*. Prentice-Hall, Englewood Cliffs, NJ, 1983.

[Mor86] A.P. Morgan. A transformation to avoid solutions at infinity for polynomial systems. *Applied Mathematics and Computation*, (18):77–86, 1986.

[Nel85] G. Nelson. Juno, a constraint-based graphic system. In *ACM Siggraph Conference Proceeding*, 1985.

[Owe91] J.C. Owen. Algebraic solution for geometry from dimen-
sional constraints. In *Proceedings of the Symposium on Solid Modeling Foundations and CAD/CAM Applications*, pages 397–407, 1991.

[Pat92] N.M. Patrikalakis. Surface-to-surface
intersections. *IEEE Computer Graphics and Applications*, 13(1):89–95, jan 1992.

[PR86] H.O. Peitgen and P.H. Richter. *The beauty of fractals, images of complex dynamical systems*. Springer Verlag, 1986.

[Sut63] I.E. Sutherland. Sketchpad, a man machine graphical communication system. In *AFIPS, Spring Joint Computer Conference*, pages 329–346, Detroit, Michigan, may 1963.

[Ver90] A. Verroust. *Etudes de problèmes liés à la définition, la visualisation et l'animation d'objets complexes en informatique graphique*. PhD thesis, Université de Paris Sud, Centre d'Orsay, 1990.

[VSR92] A. Verroust, F. Schonek, and D. Roller. Rule oriented method for parametrized computer aided design. *Computer Aided Design*, 24(3):531–540, Oct 1992.

[VVC93] J. Verschelde, P. Verlinden, and R. Cools. Homotopies exploiting newton polytopes for solving sparse polynomial systems. *SIAM J. Numerical Analysis*, 1993.

[Win88] F. Winkler. Algorithms in polynomial ideal theory and geometry. In *2nd Austrian-Hungarian Informatics Conf.*, Retzhof Austria, sep 1988.

[WMS90] C.W. Wampler, A.P. Morgan, and A.J. Sommese. Numerical continuation methods for solving polynomial systems arising in kinematics. *ASME J. on Design*, (112):59–68, 1990.

[Yak95] J.C. Yakoubsohn. An universal constant for the convergence to the newton method and application to the classical homotopy method. *Numerical Algorithms*, 1995.