Modélisation géométrique par contraintes : quelques méthodes de résolution

Samy Ait-Aoudia

ECOLE NATIONALE SUPERIEURE
DES MINES DE SAINT-ETIENNE

N° d'ordre : 109 ID

THESE

Présentée par

Samy AIT-AOUDIA

pour obtenir le titre de

DOCTEUR

DE L'UNIVERSITE DE SAINT-ETIENNE
ET DE L'ECOLE NATIONALE SUPERIEURE DES MINES DE SAINT-ETIENNE

(Spécialité : Informatique)

MODELISATION GEOMETRIQUE PAR CONTRAINTES : QUELQUES METHODES DE RESOLUTION

Soutenue à Saint-Etienne le 24 Juin 1994

COMPOSITION DU JURY

Messieurs	B. PEROCHE	Président
	R. CAUBET	Rapporteurs
	Y. GARDAN	
Madame	Y. AHRONOVITZ	Examinateurs
Messieurs	B. ARNALDI	
	D. MICHELUCCI	

ECOLE NATIONALE SUPERIEURE
DES MINES DE SAINT-ETIENNE

N° d'ordre : 109 ID

THESE

Présentée par

Samy AIT-AOUDIA

pour obtenir le titre de

DOCTEUR

DE L'UNIVERSITE DE SAINT-ETIENNE
ET DE L'ECOLE NATIONALE SUPERIEURE DES MINES DE SAINT-ETIENNE

(Spécialité : Informatique)

MODELISATION GEOMETRIQUE PAR CONTRAINTES : QUELQUES METHODES DE RESOLUTION

Soutenue à Saint-Etienne le 24 Juin 1994

COMPOSITION DU JURY

Messieurs	B. PEROCHE	Président
	R. CAUBET	Rapporteurs
	Y. GARDAN	
Madame	Y. AHRONOVITZ	Examinateurs
Messieurs	B. ARNALDI	
	D. MICHELUCCI	

REMERCIEMENTS

Je tiens ici à exprimer mes plus vifs remerciements à tous les membres du jury ainsi qu'à toutes les personnes qui, de près ou de loin, m'ont aidé durant ces années de thèse :

Monsieur Bernard Peroche, Professeur et Directeur du Département Informatique de l'Ecole Nationale Supérieure des Mines de Saint-Etienne, pour m'avoir accueilli dans son laboratoire et pour avoir bien voulu encadrer mes travaux.

Messieurs René Caubet, Professeur à l'Université Paul Sabatier de Toulouse et Yvon Gardan, Professeur à l'Université de Metz, pour avoir accepter de juger mon travail.

Madame Yolande Ahronovitz, Maître de Conférences à l'Université de Saint-Etienne et Monsieur Bruno Arnaldi, Chargé de Recherches à l'IRISA de Rennes, pour avoir accepter de faire partie du jury.

Monsieur Dominique Michelucci, Ingénieur de Recherche à l'Ecole Nationale Supérieure des Mines de Saint-Etienne, pour son aide et ses conseils éclairés.

Tous les membres passés et présents du département informatique, permanents et thésards.

Madame Marie-Line Barnéoud, irremplaçable secrétaire du département informatique, pour sa disponibilité et sa gentillesse.

Monsieur Roland Jegou, pour sa contribution à l'élaboration de l'article envoyé à Compugraphics.

Abbas Adda-Bedia, Michel Beigbeder, Hakim Benkeddad, Mohand Ourabah Benouamer, Véronique Bourgoin, Florence Dujardin, Boukhalfa Hadim, François Jaillet, Hakim Kahlouche, Hervé Lamure, Jori Léonardon, Gilles Mathieu, Jean-Luc Maillot, Abdelfattah Nahed, Nicolas Ponsi, Pascale Roudier, Hélène Sayet et Gauthier Vatant pour leur sympathique présence.

Le personnel du service de reprographie pour avoir assurer la reproduction de ce rapport.

Mes parents, mes frères et ma soeur Amina pour leur affection et leur soutien moral indéfectible.

TABLES DES MATIERES

CHAPITRE 2 :

REDUCTION DE SYSTEMES DE CONTRAINTES

CHAPITRE 3 :

SOLVEUR GEOMETRIQUE DE CONTRAINTES

INTRODUCTION

Diverses techniques de modélisation sont utilisées en synthèse d'images et en CAO (conception assistée par ordinateur) pour produire des images réalistes et analyser les propriétés géométriques des objets solides modélisés.

La modélisation géométrique a un rôle important dans le processus de conception des produits industriels. Cependant, malgré les progrés récents, la conception de formes géométriques reste donc une tâche complexe qui n'est pas aussi naturelle qu'on le voudrait.

Dans la plupart des systèmes industriels de DAO (dessin assistée par ordinateur) et de CAO, les formes géométriques, les dimensions et les positions des objets modélisés sont toutes explicitement fournies par l'utilisateur. Par exemple la conception d'une figure géométrique plane (dessin de piéces mécaniques ...) est faite par la donnée explicite de tous les constituants de celle ci (coordonnées des points, direction des droites ...).

Les objet géométriques que veut modéliser l'utilisateur doivent vérifier certaines propriétés, traditionnellement appelées contraintes. Les contraintes dans les modeleurs classiques n'avaient pas de représentation informatique et c'était à l'utilisateur de les gérer "manuellement" et d'assurer la "cohérence" en cas de modification.

Pour pallier ces inconvénients, certains systèmes de modélisation fournissent des outils de spécification des formes par des contraintes géométriques (modélisation implicite ou non-impérative). Ceci offre l'avantage de libérer l'utilisateur de la tâche fastidieuse de placement exact de ses objets. Ce type de modélisation permet, en outre, d'avoir une description claire et courte et d'assurer la mise à jour lors de la modification d'une contrainte.

Les contraintes géométriques sont des relations que vérifient les différentes parties d'un objet. On peut citer comme types de contraintes géométriques classiques en 2D pour spécifier les objets tels que les points, les droites et les cercles les contraintes de distances entre points, distances entre points et droites, parallelisme entre droites, angles entre droites, incidences entre points et droites, incidences entre points et cercles, tangeances entre cercles, tangeances entre droites et cercles .... En 3D, de nouveaux types d'objets tels que les plans introduisent de nouveaux types de contraintes tels que les angles solides.

Pour résoudre le système de contraintes, on utilise souvent des méthodes algébriques. Le principe de ces méthodes est de transformer les contraintes géométriques en équations, linéaires ou non, et de procéder ensuite à la résolution du système d'équations obtenu pour définir l'objet.

La méthode la plus utilisée est la méthode de Newton-Raphson. La matrice jacobienne sous-jacente à ces systèmes est souvent creuse. Donc la découpe du système d'équations en sous-systèmes est cruciale.

Le chapitre deux de cette thèse étudie les graphes bipartis sous-jacents à ces systèmes d'équations. Nous montrons qu'il est possible de décomposer polynomialement ces systèmes en sous-systèmes sur-contraints (plus d'équations que d'inconnues), sous-contraints (plus d'inconnues que d'équations) et bien-contraints (autant d'équations que d'inconnues) à partir du graphe biparti. Nous verrons aussi une méthode efficace de décomposition des systèmes bien-contraints en sous-systèmes irréductibles c.à.d ne pouvant plus être décomposés. Ces décompositions accélérent grandement la résolution dans le cas des grands systèmes réductibles.

Cependant, ces méthodes n'exploitent pas la "connaissance géométrique" des contraintes. Un exemple typique est donné par les constructions de pièces mécaniques qu'on peut tracer à la règle et au compas (ce qui revient à résoudre des équations du premier ou du second degré). Les méthodes algébriques ne sont pas optimales dans ces cas là. Il est plus judicieux de procéder à la résolution géométrique des contraintes et de lancer des méthodes algébriques en cas de blocage.

Nous présentons dans le chapitre trois nos travaux sur le développement d'un système de résolution de contraintes bi-dimensionnelles par une méthode géométrique. Nous étudions les différentes configurations induites par des contraintes de distances, d'angles et de tangences entre points, droites et cercles de rayon connu ou non. Ces contraintes sont représentées par un graphe. Les entités géométriques sont déterminées par un algorithme de réduction de graphes et un système à base de règles. Lorsque aucune déduction géométrique ne peut être faite, nous utilisons une méthode numérique (Bissection) pour résoudre le système de contraintes. La figure solution est donnée à un déplacement près.

CHAPITRE 1

MODELISATION PAR CONTRAINTES

INTRODUCTION.

La synthèse d'images, la CAO et le DAO utilisent diverses modélisations des solides pour les visualiser et calculer leurs propriétés géométriques et mécaniques.

Un objet peut être modélisé :

- par la donnée explicite de tous ses constituants. On parle alors de modélisation impérative. Ce type de modélisation n'assure pas automatiquement que les formes générées satisfassent certaines propriétés, que l'utilisateur souhaiterait pourtant. Ce type de modélisation peut utiliser des schémas de représentation très divers : Brep, CSG ...

- par la donnée de contraintes c.à.d d'un ensemble de relations que vérifient les différentes parties d'un objet. On parle alors de modélisation non impérative ou par contraintes. Donnons quelques exemples de contraintes : alignement de points, incidence entre un point et une droite (un plan ou toute courbe ou surface), tangences entre cercles ...

Commençons par donner quelques définitions intuitives.

- Si les contraintes données par l'utilisateur pour définir l'objet sont redondantes ou contradictoires on dira que l'objet est sur-contraint ou sur-déterminé.

• Si, par contre, ces contraintes sont insuffisantes (une infinité non dénombrable de solutions) on dira que l'objet est sous-contraint ou sous-déterminé.

On peut classer les différentes approches de modélisations en :

• méthodes algébriques :

le principe de ces méthodes est de transformer les contraintes en équations et de procéder ensuite à la résolution du système d'équations obtenu. La résolution se fait soit par des méthodes numériques soit par des méthodes symboliques.

• méthodes géométriques :

ces méthodes tentent de décomposer le problème à résoudre en petits problèmes classiques de géométrie et de combiner les solutions de ces probèmes pour avoir la solution globale. La résolution se fait par des méthodes déductives ou des algorithmes de parcours de graphes.

Une autre classification est faite dans la littérature et qui oppose modélisation variationnelle et modélisation paramétrique. Les auteurs appartiennent au champ de la CAO en mécanique. Une étude comparative détaillée est donnée par Chung et Schussel [CS 89] et Gardan et Jung [GJ 93].

• Dans les systèmes paramétriques, l'utilisateur sélectionne des ensembles prédéfinis de contraintes pour déterminer les pièces à construire. Ces systèmes résolvent une classe limitée de problèmes ; ils ne permettent pas le couplage d'équations d'ingénierie et de contraintes géométriques.

• Dans les systèmes variationnels, le logiciel n'impose aucune limitation sur les combinaisons possibles des contraintes. Contrairement aux systèmes paramétriques, ces systèmes peuvent gérer le couplage d'équations d'ingénierie et de contraintes géométriques. Une plus grande classe de problèmes peut donc être résolue.

Ce chapitre est organisé comme suit : en section A, nous décrivons différents systèmes de modélisation par contraintes. En section B, nous présentons l'approche de la théorie des graphes ou réalisabilité.

A. APPROCHES ALGEBRIQUES ET GEOMETRIQUES

1. APPROCHES ALGEBRIQUES

Le principe des méthodes algébriques est de transformer les contraintes géométriques en équations, linéaires ou non, et de procéder ensuite à la résolution du système d'équations obtenu pour définir l'objet. Ces méthodes permettent de traiter un grand nombre de types de contraintes. Il est, en outre, facile de rajouter de nouveaux types de contraintes : il suffit pour cela de rajouter les équations correspondant à cette nouvelle contrainte au système d'équations.

La résolution se fait soit :

- par des méthodes numériques (relaxation, Newton-Raphson, bissection ...)
- symboliquement (Bases de Gröbner, méthode de Wu-Ritt ...)

Nous présentons dans ce qui suit les principales méthodes de résolution et quelques logiciels de modélisation.

1.1 METHODES NUMERIQUES

La méthode numérique qui a été le plus utilisé est la méthode de Newton-Raphson. Elle a été utilisé par Lin, Gossard et Light [LGL 81], Light et Gossard [LG 82], Lee et Andews [LA 85], Nelson [Nels 85], Prusinkiewicz et Streibel [PS 86], Rocheleau et Lee [RL 87], Gossard, Zuffante et Sukurai[GZS 88], Serrano [Serr 91] et Perez et Serrano [PS 93]. Sutherland [Suth 63], Hillyard et Braid [HB 78] et Borninig [Born 79] ont utilisé la méthode de la relaxation. Barford [Barf 87] utilise une méthode de projection. Nous présentons dans cette section la méthode de relaxation, la méthode de Newton-Raphson et la méthode de la bissection qui est une variante de la méthode de Newton d'intervalles.

1.1.1 METHODE DE RELAXATION

Nous donnons dans cette section un bref aperçu de la méthode de relaxation. Cette méthode résout itérativement un système linéaire A.x = b ; un exposé plus détaillé est donné par Ciarlet [Ciar 82].

Supposons que l'on puisse écrire la matrice inversible A sous la forme d'une décomposition : A = M - N, où M est une matrice facile à inverser. On a donc les équivalences : $A.x = b \Leftrightarrow M.x = N.x + b \Leftrightarrow x = M^{-1}.N.x + M^{-1}.b$.

On associe à la dernière équation la méthode itérative :

$$x_{k+1} = M^{-1}.N.x_k + M^{-1}.b,\ k \geq 0,\ x_0 \text{ vecteur arbitraire.}$$

Soit A = (aij) une matrice d'ordre n telle que $a_{ii} \neq 0,\ 1 \leq i \leq n$, et posons

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} = D - E - F$$

(-E : sous la diagonale ; D : diagonale ; -F : au-dessus de la diagonale)

Soit ω un réel différent de zéro, on peut réécrire A telle que :

$$A = \left(\frac{D}{\omega} - E\right) - \left(\frac{1-\omega}{\omega}.D + F\right)$$

Une itération de la méthode de relaxation s'écrit

$$\left(\frac{D}{\omega} - E\right) x_{k+1} = \left(\frac{1-\omega}{\omega}.D + F\right) x_k + b,\ k \geq 0.$$

Si l'on emploie cette méthode avec un paramètre $\omega > 1$, ou $\omega < 1$, elle est dite méthode de sur-relaxation, ou de sous-relaxation, respectivement. La méthode de Gauss-Seïdel est un cas particulier de la méthode de relaxation pour $\omega = 1$.

1.1.2 METHODE DE NEWTON-RAPHSON

La méthode de Newton-Raphson résout itérativement l'équation f(x)=0, x étant un vecteur de R^n, f étant une fonction continue et dérivable. Son principe est rappelé ici ; un exposé plus approfondi est donné par Dahlquist et Bjorck [DB 74].

Supposons d'abord que n = 1. Partant d'une approximation x_0 de la solution, une séquence de valeurs pour x : x_1, x_2, x_3 ... est calculée pour approcher la solution. x_{n+1} est calculée de la façon suivante : nous calculons la tangente à la

fonction f au point $(x_n, f(x_n))$ et x_{n+1} est l'abscisse du point intersection de cette tangente et de l'axe des x (figure 1).

Pour déterminer x_{n+1}, on a donc l'équation :

$$f(x_n) + (x_{n+1} - x_n).f'(x_n) = 0.$$

La méthode de Newton-Raphson est donc défini par le schéma itératif suivant:

$$x_{n+1} = x_n + \Delta x_n, \qquad \Delta x_n = \frac{-f(x_n)}{f'(x_n)}$$

Le critère d'arrêt est de tester que la valeur absolue de Δx_n est inférieure à un epsilon donné (fixé comme marge d'erreur).

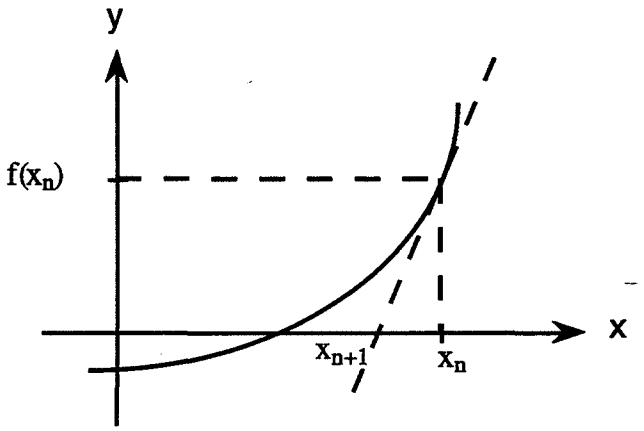


Figure 1.

Critère de convergence :

Supposons que f'(x) est différent de zéro, que f"(x) ne change pas de signe dans l'intervalle [a,b] et que f(a).f(b) < 0.

Si

$$\left|\frac{f(a)}{f'(a)}\right| < b\text{-}a \quad , \quad \left|\frac{f(b)}{f'(b)}\right| < b\text{-}a \, .$$

alors la méthode de Newton-Raphson converge de façon quadratique (ce qui signifie qu'on multiplie par deux le nombre de chiffres exacts de

l'approximation, à chaque itération) pour n'importe quelle approximation initiale x_0 appartenant à l'intervalle [a,b] (figure 2).

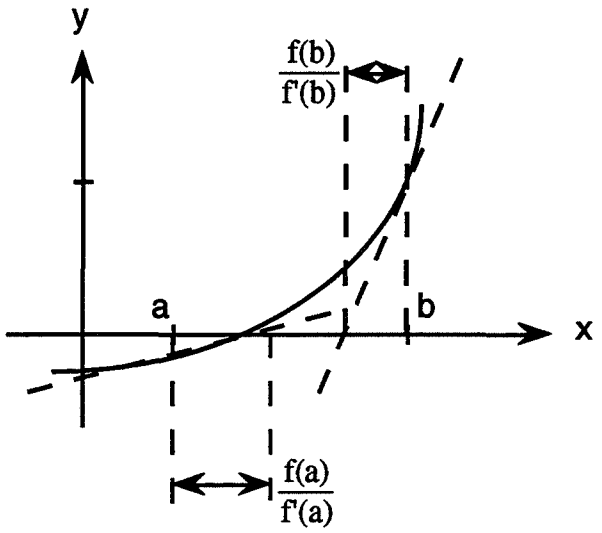


Figure 2.

La méthode de Newton-Raphson peut être généralisée à $n > 1$: soit f une application de R^n dans R^p ($p \geq 1$).

$$f'(x^{(k)}) \,.\, (x^{(k+1)} - x^{(k)}) + f(x^{(k)}) = 0 \,.$$

où

$f'(x)$ *est la matrice jacobienne de* f *(notée* J *également).*

$$f'_{ij}(x) = \frac{\partial f_i}{\partial x_j}(x) \,,\quad 1 \leq i \leq p,\ 1 \leq j \leq n \,.$$

$$x = (x_i) \,,\ 1 \leq i \leq n \,, \qquad f = (f_j) \,,\ 1 \leq j \leq p \,.$$

On procédera alors à la résolution du système linéaire d'équations suivant (à chaque itération) : $f'(x^{(k)}) \,.\, x^{(k+1)} = f'(x^{(k)}) \,.\, x^{(k)} - f(x^{(k)})$. Dans le cas où la matrice J ($J = f'(x^{(k)})$) est non singulière, ce système est résolu par des méthodes numériques directes (Gauss, Gauss-Jordan, ...) ou des méthodes itératives telles la méthode de Gauss-Seidel, de relaxation ou de Jacobi. Dans le cas où la matrice J est structurellement singulière ($n \neq p$), on peut utiliser l'itération basée sur la matrice pseudo-inverse $[J^T.J]^{-1}.J^T$ de Moore-Penrose (Golub et Van Loan [GV 87]) : $x^{(k+1)} = x^{(k)} - [J^T.J]^{-1}.J^T.\, f(x^{(k)})$.

1.1.3 METHODE DE LA BISSECTION.

La méthode de Newton-Raphson suppose connue une bonne approximation initiale : c'est là une contrainte très forte, que les méthodes par bissection permettent d'éviter. La méthode de la bissection est une variante de la méthode de Newton d'intervalles. Cette méthode est basée sur l'arithmétique d'intervalles pour encadrer toutes les solutions dans un domaine donné. On la rappelle très succinctement ici. Une théorie complète peut être trouvée dans [Moor 66].

1.1.3.1 ARITHMETIQUE D'INTERVALLES.

On présentera dans ce qui suit quelques rappels sur l'arithmétique d'intervalles.

Définitions :

Un intervalle est un ensemble fermé borné de nombres réels (en pratique des nombres flottants).

$$[a,b] = \{x : a \leq x \leq b\}.$$

Un vecteur d'intervalles est un nuplet ordonné d'intervalles.

$$V = (X_1, X_2, \ldots, X_n) \qquad \text{où} \qquad X_i = [a_i, b_i].$$

Ceci correspond à une boite géométrique de dimension n.

Soient les deux intervalles $X_1 = [a_1, b_1]$ et $X_2 = [a_2, b_2]$ et * l'opération binaire appartenant à l'ensemble {+, -, ., /}, on peut définir $X_1 * X_2$ comme suit :

$X_1 * X_2 = \{x*y : x \text{ appartient à } X_1 \text{ et } y \text{ appartient à } X_2\}$.

Cette définition donne les quatre cas suivants :

$X_1 + X_2 = [a_1 + a_2, b_1 + b_2]$.

$X_1 - X_2 = [a_1 - b_2, b_1 - a_2]$.

$X_1 . X_2 = [\min(a_1.a_2, a_1.b_2, b_1.a_2, b_1.b_2), \max(a_1.a_2, a_1.b_2, b_1.a_2, b_1.b_2)]$.

$1 / X_1 = [1 / b_1, 1 / a_1]$ si 0 n'appartient pas à X_1.

Si 0 appartient à X_1 alors $1 / X_1$ sera égal à :

$[-\infty, 1/a_1]$ si $b_1 = 0$.

$[-\infty, 1/a_1] \cup [1/b_1, +\infty]$ si $a_1 < 0$ et $b_1 > 0$.

$[1/b_1, +\infty]$ si $a_1 = 0$.

Soit $X = [a,b]$, on définit le milieu de X par $m(X) = (a + b) / 2$ et la largeur de X par $l(X) = b - a$. De même, pour un vecteur d'intervalles $V = (X_1, X_2, ..., X_n)$, on définit le milieu de V par $m(V) = (m(X_1), m(X_2), ..., m(X_n))$ et la largeur de V par $l(V) = (l(X_1), l(X_2), ..., l(X_n))$. Un vecteur d'intervalles est dit symétrique s'il peut être exprimé de la façon suivante :

$V = r.[-1, 1]$ où r est un vecteur de réels.

Soit f une fonction à n variables réelles $x_1, x_2, ..., x_n$. On définit une extension d'intervalles de f notée F comme une fonction ayant pour arguments n variables d'intervalles. $F(x_1, x_2, ..., x_n)$ sera égale à $f(x_1, x_2, ..., x_n)$ pour les arguments réels. La fonction F convergera vers la fonction f quand la largeur des intervalles arguments de F diminuera.

1.1.3.2. ALGORITHME.

Soient

$f = (f_1, ... , f_n)$. f_i fonction non linéaire.

$x = (x_1, ... , x_n)$. vecteur de variables.

On recherche les encadrements des solutions de l'équation :

$$f(x) = 0 . \qquad (1)$$

Moore [Moor 66] fut le premier à utiliser une méthode analytique par intervalles pour encadrer une solution x^* de l'équation (1).

Soit x_0 une solution approchée. En utilisant le théorème de Taylor et en développant $f(x^*)$ en x_0 on définit ß tel que :

$$f(x_0) + J(ß).(x^* - x_0) = f(x^*) = 0 . \qquad (2)$$

où

J(ß) est le jacobien évalué au point ß.

D'après Moore, si X est un vecteur d'intervalles contenant x_0 et x^* alors ß appartient à X. Il remplace dans l'équation (2) J(ß) par la matrice d'intervalles J(X). x_0 est pris comme étant le milieu du domaine X (x_0 sera noté x_m).

Pour éviter de calculer l'inverse d'une matrice d'intervalles, on multiplie l'équation (2) par M qui est une inversion approximative de J_c, J_c étant J(m(X)). En fait M est une matrice arbitraire non singulière. On utilisera ensuite une procédure d'élimination de Gauss.

On réécrit (2) comme :

$$M.f(x_m) + M.J(X).(x^* - x_m) = 0 . \qquad (3)$$

Les produits $M.f(x_m)$ et M.J(X) sont calculés avec une arithmétique d'intervalles.

Krawczyk [Kraw 69] introduit une variante de la méthode d'intervalles de Newton pour éviter la procédure d'élimination de Gauss d'une matrice d'intervalle en ne cherchant pas à calculer une solution exacte de (3). Il calcule la boite K(X) (K(X) est connu sous le nom "d'opérateur de Krawczyk-Moore") :

$$K(X) = x_m - M.f(x_m) + [I - M.J(X)].[X - x_m] . \qquad (4)$$

On peut retrouver ce résultat en utilisant le théorème de Ortega et Rheinboldt [OR 70] et l'algorithme de Newton.

Thèorème [OR 70] :

si x_0 et x_1 appartiennent au domaine X alors $F(x_0) - F(x_1)$ appartient à $F'(X).(x_0 - x_1)$.

Soit $N(x) = x - M.f(x)$ l'image de x appartenant à X par l'algorithme de Newton.

$N'(X) = I - M.F'(X)$ où I est la matrice identité.

$N(x) - N(x_m) \in N'(X).(x - x_m)$,

$N(x) \in N(x_m) + (I - M.F'(X)).(x - x_m)$,

$N(x) \in x_m - M.f(x_m) + (I - M.F'(X)).(x - x_m)$.

Dans tous les calculs, un nombre réel x_m est considéré comme un intervalle dégénéré $[x_m, x_m]$.

Si une solution de (1) est contenue dans X alors cette solution appartient à la boite K(X). Puisque K(X) n'est pas forcément contenue dans X on a le schéma itératif suivant :

$X^{(i+1)} = X^{(i)} \cap K(X^{(i)})$.

X^0 domaine initial donné .

D'où la procédure suivante :

Pour chaque boite X

Calculer K(X).

Si $(K(X) \cap X) = \varnothing$

Alors X ne contient pas de racines.

Si $(K(X) \cap X) \neq \varnothing$

$r = ||I - M.J(X)||$.

Si $(K(X) \subset X)$

Alors

Si $r < 1$

Alors Le schéma de Newton simple $x_{n+1} = p(x_n)$ va converger vers une solution unique quelle que soit la valeur de départ x_0 appartenant à X.

Sinon Subdiviser K(X).

Sinon

soit a la moitié de la largeur minimum des intervalles de X.

Si $r < 1$ et $||m(X) - p(m(X))|| < (1 - r).a$

Alors Le schéma de Newton simple $x_{n+1} = p(x_n)$ va converger vers une solution unique quelle que soit la valeur de départ x_0 appartenant à B(m(X),a).

Sinon Subdiviser $(K(X) \cap X)$

où

m(X) est le vecteur de réels milieu de X.
B(m(X),a) = {x appartenant à X : $\| x - m(X) \| \leq a$ }.
p(x) = x - M.f(x).

Les normes utilisées sont les suivantes :

La norme d'un intervalle est le maximum des valeurs absolues des deux bornes.

La norme d'un vecteur d'intervalles est le maximum des normes des différents intervalles constituants ce vecteur.

La norme d'une matrice d'intervalles est le maximum des sommes des normes des éléments des vecteurs lignes de cette matrice.

La méthode de la bissection permet de trouver toutes les solutions d'un système d'équations non linéaires dans un domaine donné.

L'avantage de cette méthode est donc de permettre de trouver toutes les figures géométriques satisfaisant un ensemble de contraintes. Cette méthode est, par contre, plus coûteuse en temps de calcul que la méthode de Newton-Raphson. Un autre inconvénient, qui ne survient pas souvent, surgit lorsqu'une solution appartient à plusieurs intervalles après une subdivision : on aboutit à une solution de multiplicité dépendant du nombre de subdivisions alors qu'on a réellement une seule solution [MJ 79] ; de plus il y a un ralentissement de la méthode dans ce cas de figure.

Une méthode de bissection, que nous n'avons pas eu le temps d'implanter et de tester, a été proposée fin 93 par Dedieu et Yakoubsohn [DY 93]. En fait, là est le principal inconvénient de l'approche numérique : la résolution des systèmes algébriques est encore un sujet très actif de recherches, on ne dispose donc pas à l'heure actuelle de la solution définitive, ou, au moins, d'une solution efficace.

1.2. METHODES SYMBOLIQUES

Ces méthodes (Gosling [Gosl 83], Ericson et Yap [EY 88], Kondo [Kond 92], Dhingra et Mani [DM 93] et Buchanan et Pennington [BP 93]) résolvent symboliquement un système d'équations qui doivent être polynomiales. Elles sont plus générales que les méthodes numériques. Les systèmes de modélisations ayant un solveur symbolique de contraintes utilisent des outils de calcul algébrique tels que MACSYMA, REDUCE et MAPLE. Le noyau de ces systèmes est l'algorithme de Buchberger [Buch 65] qui calcule les bases de Gröbner. Ces solveurs sont toutefois très lents et consomment beaucoup de place mémoire. D'après Lazard [Laza 92], le calcul des bases de Gröbner de systèmes de plus de dix inconnues est sans espoir. Les bases de Gröbner peuvent cependant être utilisées pour résoudre des problèmes élémentaires, courants ; on en déduit des procédures numériques. Cette approche a déjà été utilisée par Hanrahan [Hanr 83] pour engendrer des procédures efficaces d'intersection entre des rayons et des surfaces algébriques. Nous présentons dans ce qui suit l'algorithme de calcul des bases de Gröbner. Un exposé détaillé est donné par Winkler, Bucheberger et Lichtenberger [WBL 85], Kutzler [Kutz 87], Chou, Schelter et Yang [CSY 87] et Winkler [Wink 88].

1.2.1 BASES DE GROBNER

1.2.1.1 TERMINOLOGIE

Soit K un corps et $K[y_1,...,y_n]$ l'anneau des polynômes en y_i et à coefficients dans K (En pratique, K est très souvent le corps des rationnels ou des extensions de ce corps). On utilisera les notations suivantes :

f,g et h pour désigner des polynômes dans $K[y_1,...,y_n]$,
F et G sont des ensembles finis de $K[y_1,...,y_n]$,
t et u sont des produits de puissance de la forme y_i^j,
a et b sont des éléments de K et
i et j sont des nombres entiers.

Définition :

L'idéal généré par F est :

$$Idéal_{K[y1,...,yn]}(F) = \{\Sigma h_i f_i : f_i \in F,\ h_i \in K[y_1,...,y_n]\}.$$

Intuitivement, si F est vu comme un ensemble de contraintes vérifiées par les y_i, l'idéal est l'ensemble de toutes les propriétés vérifiées par les y_i. Les bases de Gröbner sont des représentations canoniques de ces idéaux.

On aura besoin d'un ordre < total sur les produits de puissance ou monômes. Cet ordre doit être "compatible", i.e préservé par la multiplication : si $m_1 < m_2$ alors $m.m_1 < m.m_2$, quelque soit le monôme m. Des exemples d'un tel ordre, dans la cas de deux variables, sont donnés par ce qui suit :

- l'ordre lexicographique pur ($1 < y_1 < y_2 < y_1^2 < y_1y_2 < y_2^2 < y_1^3 < \dots$)
- l'ordre total des degrés ($1 < y_1 < y_1^2 < \dots < y_2 < y_1y_2 < y_1^2y_2 < \dots < y_2^2 < y_1y_2^2 < \dots$)

Définition :

- Un polynôme g est réduit à h modulo f ($f \neq 0$) si il existe u,b, $b \neq 0$ tels que :

$$h = g - b.u.f.$$

Les notations suivantes seront utilisées par la suite :

coef(g,t) désigne le coefficient de t dans g selon l'ordre <,
lpp(g) désigne le plus grand produit de puissance selon l'ordre <,
lc(g) désigne le coefficient de lpp(g) selon l'ordre <,
$g \Rightarrow_f h$ (g est réduit à h modulo f),
$g \Rightarrow_F h$ (g est réduit à h modulo F) ssi il existe $f \in F$ tel que $g \Rightarrow_f h$,
$\Rightarrow_f^+$ ($\Rightarrow_F^+$) désigne la fermeture transitive de $\Rightarrow_f$ ($\Rightarrow_F$),
$\Rightarrow_f^*$ ($\Rightarrow_F^*$) désigne la fermeture réflexive et transitive de $\Rightarrow_f$ ($\Rightarrow_F$),
$\Leftrightarrow_f^*$ ($\Leftrightarrow_F^*$) désigne la fermeture réflexive, symmetrique et transitive de $\Rightarrow_f$ ($\Rightarrow_F$).

Définitions :

- *h est irréductible modulo F si il n'existe pas h tel que* $g \Rightarrow_F h$.
- *h est une forme normale de g modulo F ssi* $g \Rightarrow_F^* h$ *et h est irréductible modulo F.*
- *Un algorithme est de forme normale si pour chaque entrée g et F il calcule la forme normale de g modulo F.*

1.2.1.2 DEFINITION DES BASES DE GROBNER

La forme normale d'un polynôme g modulo une base F n'est pas nécessairement unique. Il peut exister plusieurs réductions. Les bases de Gröbner sont les bases où ceci ne peut pas arriver.

Définition :

F est appelée base de Gröbner si et seulement si chaque polynôme g a une forme normale unique modulo F.

Théorème : (Caractérisation des bases de Gröbner)

Soit Nf un algorithme arbitraire de forme normale. F est une base de Gröbner si et seulement si pour tous les f et g on a : $f \Leftrightarrow_F^* g$ ssi Nf(f,F) = Nf(g,F).

1.2.1.3 CONSTRUCTION DES BASES DE GROBNER

Les bases de Gröbner sont utilisées pour résoudre beaucoup de problèmes dans différents domaines. Ces problèmes, pour un ensemble arbitraire F, sont résolus en transformant F en une base de Gröbner G et en utilisant G pour ces mêmes problèmes. Le problème est exprimé par :

Donnée : F

Trouver : G telle que Idéal (G) = Idéal (F) et G est une base de Gröbner.

Avant d'énoncer l'algorithme de Buchberger une définition des concepts de "complétude" et de "paires critiques" est donnée : chaque fois que la réduction d'un polynôme donne deux formes normales modulo une base arbitraire F, il suffit de rajouter leur différence à cette base pour obtenir une forme normale unique. Ceci est la définition de la "complétude" de la base. Cette méthode n'est pas praticable car un nombre infini de polynômes doit être testé. Buchberger a remarqué qu'il suffit de tester un ensemble fini de solutions qu'il appelle S-polynômes. Ces polynômes sont construits à partir de toutes les paires d'éléments de la base. Chaque paire est appelée "paire critique".

Définition :

Soient f_1 et f_2 deux polynômes. On appelle S-polynôme de f_1 et f_2 :

S-polynôme $(f_1, f_2) = u_1.f_1 - (lc(f_1) / lc(f_2)).u_2.f_2$

où

u1 et u2 sont tels que $u_1.f_1$ et $u_2.f_2$ ont le même plus grand produit de puissance.

Un S-polynôme a zéro comme forme normale car il appartient à l'idéal généré par F. L'algorithme de Buchberger [Buch 65] est donné dans ce qui suit. La preuve est donnée dans [Buch 70].

Algorithme :

G := F ;
B := {$(f_1,f_2) : f_1,f_2 \in G, f_1 \neq f_2$} ;
Tant que B ≠ ∅ **faire**
    (f_1,f_2) := une paire de B ;
    B := B - {(f_1,f_2)} ;
    h := forme normale du S-polynôme (f_1,f_2) modulo G ;
    si h ≠ 0 **alors** G := G ∪ {h} ; B := B ∪ {(g,h) : g ∈ G} **fsi** ;
fait ;

Une base de Gröbner ainsi définie n'est pas nécessairement unique. Il est possible de réduire un polynôme de la base modulo les autres polynômes de la base. Ceci conduit à la définition suivante :

Définition :

G est une base de Gröbner réduite si et seulement si
G est une base de Gröbner et pour tout $g \in G$:
g est irréductible modulo G - {g} et lc(g) = 1.

Partant d'une base de Gröbner arbitraire G, il est possible de construire la base de Gröbner réduite pour le même idéal en réduisant successivement tous les polynômes de G modulo les autres polynômes de G et en normalisant les coefficients du plus grand produit de puissance à 1. Buchberger dans [Buch 79]

et [Buch 85] donne plusieurs améliorations de l'algorithme initial, en particulier des critères évitant la prise en compte de certaines paires critiques.

Exemple :

Soit l'ensemble $F = \{-y3 + y1y2 + y1^2, y2y3 - y2^2 + y1y2, -y3^2 + y2^2 + y1^2\}$.

La base de Gröbner réduite G de F en utilisant l'ordre lexicographique pur est la suivante :

$$
\begin{aligned}
G = \{ \; & y1^6 + y1^5 - y1^4 - y1^3, \\
& y1y2 - 3/2.y1^5 - y1^4 + 3/2.y1^3 + y1^2, \\
& y2^2 - y1^5 - y1^4 + y1^3 + y1^2, \\
& y3 - 3/2.y1^5 - y1^4 + 3/2.y1^3 \; \}.
\end{aligned}
$$

La résolution des systèmes d'équations polynomiales se fait par la transformation de ce système en une base de Gröbner réduite en utilisant l'ordre lexicographique pur*. Cette base est de forme triangulaire. Il faut résoudre la première équation à une inconnue et propager ensuite ces valeurs. A chaque étape on ne résout donc que des équations polynomiales à une inconnue. Diverses méthodes peuvent être utilisées pour résoudre ces équations : les suites de Sturm, la méthode de Bairstow ou la méthode QR (voir [DB 74] et [GV 87]).

Outre la résolution des systèmes d'équations, les bases de Gröbner peuvent être utilisées pour la démonstration automatique de théorèmes en géométrie. Le principe de base est de transformer un problème géométrique donné en un problème algébrique : les hypothèses et la conclusion sont exprimées par des équations polynomiales en choisissant un système de coordonnées approprié.

Exemple :

Soit le triangle ABC illustré par la figure 3. La médiane issue de C coupe le segment AB en M ; la droite issue de A et médiane à CM coupe la droite BC en N. Prouver que la longueur du segment NB est le double de la longueur du segment NC.

---

* En fait, il existe une méthode plus rapide : calculer une base de Gröbner pour l'ordre total des degrés, et la convertir en une base de Gröbner pour l'ordre lexicographique pur en résolvant un système linéaire [Sene 90].

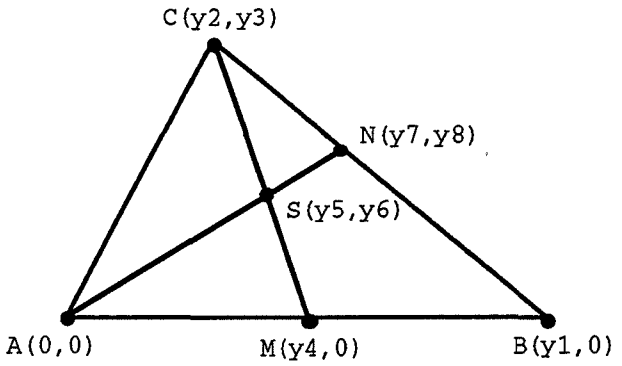


Figure 3.

On choisit un système de coordonnées de telle sorte que A soit à l'origine et que B appartienne à l'axe des abscisses. La formulation algébrique du problème posé est la suivante :

hypothèses

M milieu de AB :	$h_1 = 2y_4 - y_1 = 0$,
S milieu de MC :	$h_2 = 2y_5 - y_2 - y_4 = 0$ et $h_3 = 2y_6 - y_3 = 0$,
N appartient à AS :	$h_4 = y_5y_8 - y_6y_7 = 0$,
N appartient à BC :	$h_5 = (y_2 - y_1)y_8 - y_3(y_7 - y_1) = 0$,

conjecture

longueur NB = double longueur NC :

$$c = 4((y_7 - y_2)^2 + (y_8 - y_3)^2) - (y_7 - y_1)^2 - y_8^2 = 0.$$

L'idée maîtresse est de vérifier que le polynôme conclusion appartient à l'idéal des polynômes hypothèses.

Pour qu'un théorème soit valide, des conditions de non dégénérescence doivent être vérifiées. Il est humainement très difficile de ne pas oublier de telles dégénérescences. Aussi fait-on en sorte que ce soit le logiciel qui les détecte. Ces dégénérescences sont exprimées par un polynome s qui doit être non nul. Ce polynome s est appelé la condition subsidiaire ; il est calculé par l'algorithme suivant :

Algorithme

H:= $\{h_1, \ldots, h_m\}$;

G:= Base de Gröbner de H ∪ {c.y' - 1};
/* y' est une nouvelle variable */
si 1 ∈ G
alors s:= 1;
sinon **pour_tout** g ∈ G ∩ Q[y_1,..., y_m]
faire **si** 1 ∉ Base de Gröbner de H ∪ {g.y' - 1} **alors** s:= g **fsi**;
fait
fsi;

Cet algorithme appliqué au problème précédent aboutit au polynome s = y_1y_3. La conjecture est donc vraie pour tout les triangles non dégénérés ($y_1 \neq 0$ et $y_3 \neq 0$). On peut ainsi redémontrer tous les théorèmes de la géométrie euclidienne : théorèmes sur les triangles, théorème de Morley (les trois points d'intersection des trisecteurs adjacents d'un triangle forment un triangle équilatéral), théorème de Simson, ... [CSY 87] et [Carr 89].

1.3 LOGICIELS DE MODELISATION

Nous présentons dans cette section quelques logiciels de modélisation basés sur une méthode algébrique de résolution du système d'équations issu des ontraintes.

1.3.1 RESOLUTION NUMERIQUE

Nous décrivons dans ce qui suit les logiciels développés par Sutherland [Suth 63], Borning [Born 79], Light et Gossard [LG 82], Lee et Andrews [LA 85], Nelson [Nels 85], Rocheleau et Lee [RL 87], Serrano [Serr 91] et Sridhar, Agrawal et Kinzel [SAK 93].

• Sutherland [Suth 63] a développé SKETCHPAD un logiciel de dessin permettant la définition de différents objets et contraintes. Il a été le précurseur dans le domaine des systèmes contraints.

SKETCHPAD permet à un utilisateur de définir de nouveaux types de dessins en combinant les types de base (point, segments de droite et arcs de cercle) et

d'autres dessins prédéfinis par l'utilisateur. Ces définitions de dessins peuvent être utilisées de deux façons :

- l'utilisateur peut utiliser une copie de la définition d'un dessin pour la composition d'autres dessins. Aucun lien n'est maintenu entre la copie et le dessin original. L'utilisateur peut modifier la copie.

- La définition d'un dessin peut être utilisée comme "maître" pour générer différentes instances de ce dessin. Chaque instance a la même forme que le "maître". Les modifications sur le dessin "maître" sont répercutées sur toutes ses instances.

Les contraintes sont utilisées pour spécifier les conditions qu'un dessin doit satisfaire. Le système SKETCHPAD comprend un mécanisme de satisfaction de contraintes utilisant la propagation. Cette phase est précédée d'une étape de planification appliquant un tri topologique pour ordonner les contraintes. Si aucun ordre ne peut être trouvé, la relaxation est utilisée. La relaxation est une technique numérique classique d'approximation trouvant les solutions d'un système d'équations. Cette méthode calcule pour chaque équation une estimation de l'erreur induite par l'assignation de valeurs particulières aux variables. La relaxation tente de minimiser l'erreur globale (somme des erreurs de chaque équation) en perturbant les valeurs assignées aux variables. La relaxation réussit si l'erreur tend vers zéro. Malheureusement, cette méthode est assez lente, et surtout, elle peut diverger.

• Borning [Born 79] a développé le logiciel THINGLAB. Ce logiciel est un produit de simulation. Il fournit un environnement pour construire des modèles dynamiques d'expérimentation en géométrie, physique... e.g simulations d'objets géométriques contraints, circuits électriques, liens mécaniques.

Ce système est basé sur SKETCHPAD et SMALLTALK [Inga 78]. La factorisation orientée objet est un outil d'organisation de base. L'héritage et la composition sont utilisées pour décrire la structure de la simulation.

Dans THINGLAB chaque contrainte comprend une règle et un ensemble de méthodes qui peuvent être invoquées pour satisfaire cette contrainte. La règle est utilisée par le système pour tester si la contrainte est satisfaite ou non.

L'ensemble des contraintes sur un objet peut être incomplet, circulaire ou contradictoire. C'est le système qui se charge de la résolution.

Un exemple de contrainte et ses méthodes associées est donné par la classe point_milieu_droite suivante :

Classe point_milieu_droite

SuperClasse

Objet_géométrique

Champs

ligne type droite

point_milieu type milieu

Contraintes

(ligne.point1 + ligne.point2) / 2 = point_milieu

point_milieu ← (ligne.point1 + ligne.point2) / 2

ligne.point1 ← ligne.point2 + ((point_milieu - ligne.point2) * 2)

ligne.point2 ← ligne.point1 + ((point_milieu - ligne.point1) * 2)

Dans cet exemple, trois méthodes peuvent être utilisées pour resatisfaire la contrainte après une modification : une méthode change le milieu du segment et les deux autres méthodes changent les points extrémités de ce segment. Si aucun des points point_milieu, ligne.point1 et ligne.point2 n'est ancré (i.e fixé) le système sélectionne la première méthode pour la satisfaction. Si par contre un point est fixé le choix de la méthode est fait en conséquence : si par exemple point_milieu est fixé c'est la deuxième méthode qui est utilisée pour satisfaire la contrainte.

Partant de la remarque suivante : les valeurs des objets changent plus fréquemment que les structures, l'auteur essaye d'optimiser le processus de satisfaction. Par exemple si l'utilisateur déplace un point à l'aide de la souris, les coordonnées de ce point changent à chaque rafraîchissement de l'écran. Chaque fois qu'une valeur est changée par l'utilisateur, d'autres valeurs sont mises à jour par le logiciel pour maintenir la validité des contraintes. Un plan

de satisfaction est établi et compilé. A chaque modification, c'est cette méthode précompilée qui est invoquée pour la satisfaction des contraintes.

Pour établir le plan de satisfaction de contraintes THINGLAB utilise, dans l'ordre les trois techniques suivantes : propagation de degrés de liberté, propagation d'états connus et relaxation.

Dans la propagation de degrés de liberté, le solveur de contraintes vérifie si un objet considéré a assez de degrés de liberté pour être altéré de façon à satisfaire ses contraintes. On fait de même pour tous les autres objets jusqu'à ce que toutes les contraintes soient satisfaites ou qu'il n'y ait plus de degrés de liberté. Dans la deuxième technique appelée "propagation d'états connus" par Borning, le solveur tente de trouver un ordre de résolution se basant sur les dépendances fonctionnelles entre les objets connus et les objets à déterminer. Si les deux méthodes précédentes échouent, le solveur a recours à la technique de relaxation qui permet de gérer le problème de circularité. Dans cette dernière technique, les contraintes doivent être approximées par des équations linéaires. THINGLAB a été étendue pour tenir compte de contraintes temporelles par Borning et Duisberg [BD 86] et Duisberg [Duis 86] ; le nouveau système est appelé ANIMUS.

• Light et Gossard [LG 82] ont développé un système de CFAO basé sur des modèles géométriques. Un objet est déterminé par la donnée d'un ensemble de contraintes de dimension. Ces contraintes concernent les points caractéristiques de l'objet considéré. Chaque contrainte est décrite par une équation non-linéaire. La forme de l'objet est déduite de la résolution simultanée par la méthode de Newton-Raphson de l'ensemble des équations.

Light et Gossard étudient les cas de schémas de contraintes sur- et sous-déterminés. La matrice jacobienne est structurellement singulière si le nombre m d'équations (contraintes) est différent du nombre n d'inconnues (coordonnées des points caractéristiques). Si m = n la matrice jacobienne peut être numériquement singulière. Ceci arrive lorsqu'une partie d'un schéma de contraintes est sur-déterminée et une autre partie de ce même schéma est sous-déterminée. La non-singularité de la matrice jacobienne est donc une condition nécessaire et suffisante pour la validité d'un schéma contraint. Les auteurs utilisent une version modifiée de la méthode de Doolittle [DB 74] pour résoudre le

système d'équations quand la matrice jacobienne est singulière. Cependant, selon Rocheleau et Lee [RL 87], cette méthode est assez lente. Le chapitre deux de cette thèse présentera une méthode efficace de détection des cas sous- ou sur-contraints.

• Lee et Andrews [LA 85] ont développé un logiciel de modélisation par contraintes. Une technique est proposée pour permettre à un utilisateur de créer un assemblage d'objets en spécifiant uniquement les contraintes entre ces différents objets. Les contraintes utilisées sont les suivantes :

Collage (against) : face contre face.
Colinéarité (fit).

La contrainte de collage est illustrée par la figure 4. Chaque face contrainte est définie par un vecteur normal et un point sur cette face. Pour satisfaire la contrainte de collage, les normales des deux faces doivent être parallèles et de directions opposées et les points de ces faces doivent appartenir à un même plan. La contrainte de colinéarité est illustrée par la figure 5. Pour satisfaire cette contrainte, il faut qu'une ligne centrale du premier objet soit colinéaire à une ligne centrale du second objet.

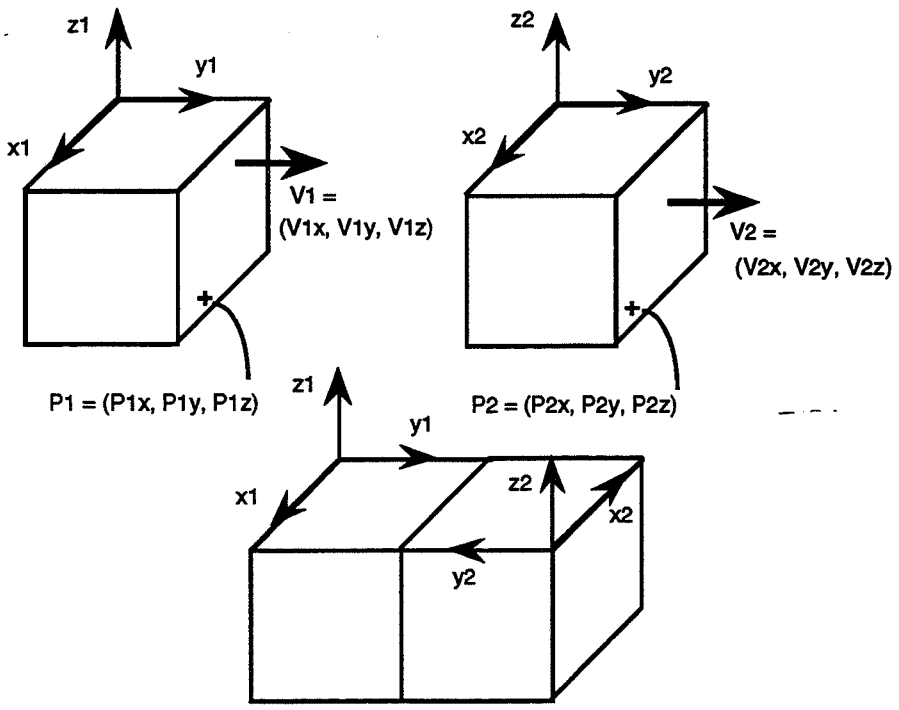


Figure 4.

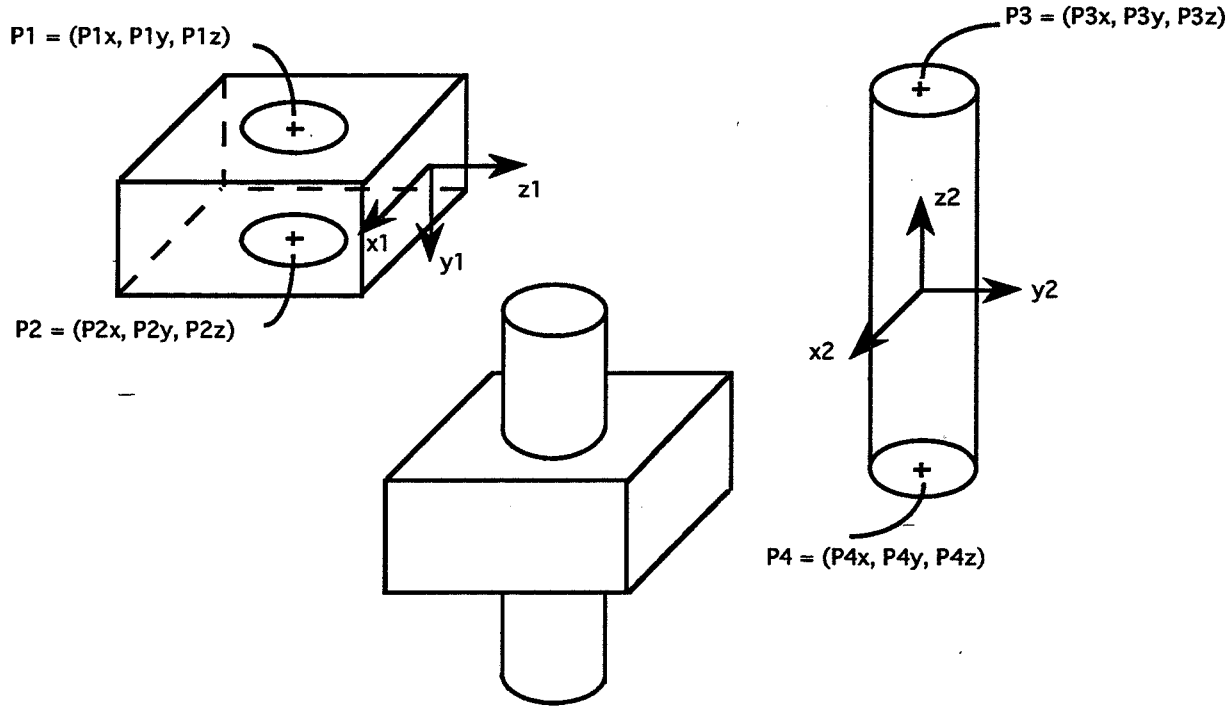


Figure 5.

Les auteurs proposent une méthode de calcul des douze variables (neuf valeurs de rotations et trois valeurs de translation) de la matrice de transformation permettant de placer un objet relativement à un autre. Cette matrice est décrite par les relations spatiales définies par les conditions de collage et de colinéarité. Donc pour assembler N objets, N-1 matrices de transformations sont nécessaires (un objet étant fixé au départ). On aboutit finalement à un système d'équations à 12.(N-1) variables à résoudre. Dans cette méthode le nombre d'équations est supérieur ou égal au nombre d'inconnues. Les auteurs utilisent l'algorithme décrit par Light et Gossard [LG 82] pour éliminer les équations redondantes. Le système d'équations est alors résolu par la méthode de Newton-Raphson. Le choix des valeurs initiales n'est pas indiqué.

• Nelson [Nels 85] a développé JUNO, un logiciel de construction de dessins plans. Les contraintes géométriques sont spécifiées de façon déclarative. Cette spécification est similaire à celle de METAFONT [Knut 79]. Cependant METAFONT ne prend en considération que les contraintes correspondant à des équations linéaires. JUNO translate les contraintes géométriques en équations

linéaires ou non et résout le système d'équations par la méthode de Newton-Raphson.

La syntaxe des commandes de JUNO est la suivante :

LET Variables | Contraintes **IN** Commande **END**

où

Variables est une liste de variables locales à introduire,
Contraintes est une conjonction de contraintes portant sur les variables,
Commande est la procédure à exécuter.

Les quatre types de contraintes permises par JUNO sont les suivantes :

(x, y) **CONG** (u, v)	La distance de x à y est égale à la distance de u à v.
(x, y) **PARA** (u, v)	xy est parallèle à uv.
HOR (x, y)	La direction de x à y est horizontale.
VER (x, y)	La direction de x à y est horizontale.

Les commandes utilisées par JUNO sont définies par une liste de points indépendants, une liste de points dépendants, un ensemble de contraintes et des motifs de remplissage. Les valeurs initiales pour la méthode de résolution de Newton-Raphson sont tirées de l'esquisse du dessin. Mais, même dans ce cas cette méthode peut diverger ou converger vers une solution non voulue.

• Rocheleau et Lee [RL 87] reprennent les travaux menés par Lee et Andrews [LA 85] en modifiant la procédure d'élimination des équations redondantes. Les auteurs constatent que dans l'approche précédente la majorité du temps de calcul est utilisé par la routine de Light et Gossard [LG 82] recherchant les équations redondantes. Les auteurs proposent d'utiliser la méthode des moindres carrés pour ne plus s'occuper des équations redondantes (ces équations sont implicitement traitées par la méthode utilisée).

Le nombre m d'équations $(f_1, f_2, \ldots\ldots, f_m = 0)$ est supérieur au nombre d'inconnues $(x_1, x_2, \ldots\ldots, x_n)$. Alors que la méthode de Newton-Raphson classique est basée sur l'itération suivante :

$$f'(x^{(k)}) \,.\, (x^{(k+1)} - x^{(k)}) + f(x^{(k)}) = 0 \,.$$

$$[J] \,.\, \{\Delta x\} = \{R\}.$$

La méthode des moindres carrés est basée sur l'itération :

$$[J^T.J] \,.\, \{\Delta x\} = [J^T] \,.\, \{R\} \,.$$

donc

$$\{\Delta x\} = [J^T.J]^{-1}.\, [J^T] \,.\, \{R\} \,.$$

Cette méthode n'est autre que la méthode utilisant la matrice pseudo-inverse de Moore-Penrose.

• Serrano [Serr 91] propose un algorithme de résolution du système de contraintes basé sur les graphes bipartis sous-jacents à ces systèmes. Cet algorithme s'applique aussi bien au cas bi-dimensionnel qu'au cas tri-dimensionnel. Le graphe biparti, associé à un système d'équations, posséde un sommet par équation, un sommet par inconnue et une arête entre une inconnue x et une équation y si, et seulement si, x apparait dans l'équation y.

L'auteur constate qu'il existe une correspondance directe entre le couplage du graphe biparti et l'objet conçu. Le couplage n'étant pas unique, il est alors possible d'obtenir plusieurs formes pour l'objet conçu dépendant du couplage initial. Donc le fait qu'un couplage n'aboutit pas à un objet consistant n'implique pas qu'il n'en existe pas. Une méthode qui génére tous les couplages possibles [IRT 78] est nécessaire selon l'auteur.

L'algorithme décrit trouve une combinaison des contraintes qui aboutit à un objet consistant. Quand une composante fortement connexe du graphe biparti est trouvée, il est inutile de considérer les différents couplages à l'intérieur de cette composante car les contraintes considérées sont à résoudre simultanément. La méthode de Newton-Raphson est utilisée pour résoudre ce système d'équations simultanées.

L'auteur propose d'étudier d'autres règles pour réduire l'explosion des combinaisons de couplages. Le chapitre deux considère le même graphe que Serrano, mais propose des méthodes plus efficaces.

• Sridhar, Agrawal et Kinzel [SAK 93] présentent un algorithme de décomposition et résolution du système de contraintes se basant sur une méthode matricielle. Cet algorithme traite aussi bien le problème plan que le problème tridimensionnel.

Les auteurs supposent que le problème est modélisé mathématiquement par m équations non redondantes et non conflictuelles à n inconnues, n étant supérieur à m. Les conceptions sur-contraintes ne sont pas traitées par cet algorithme. La structure de l'ensemble des équations algébriques est décrite par une matrice booléénne que les auteurs appellent matrice-occurence.

Soit un système sous-contraint de m contraintes (équations) à n variables ($m < n$). Les éléments de la matrice-occurence A, qui est une matrice mxn, sont définis par les relations suivantes :

$A_{ji} = 1$ si la variable x_i apparait dans l'équation f_j.
$A_{ji} = 0$ si la variable x_i n'apparait pas dans l'équation f_j.

La matrice A est rectangulaire. Les lignes de A correspondent aux équations et les variables sont représentées par les colonnes. Bien sûr, cette représentation n'est rien d'autres que la représentation matricielle du graphe biparti considéré par Serrano.

Exemple :

Soient les équations non-linéaires suivantes représentant un schéma de conception :

$$f_1 : x_1^2 - x_2 = 0.$$
$$f_2 : x_1 + x_2^3 + 4.x_3 - 2.x_4^2 = 0.$$
$$f_3 : 3.x_3^4 - x_4 = 0.$$
$$f_4 : x_5^3 + 5.x_6^2 = 0.$$

La matrice-occurence correspondant au système d'équations précédent est la matrice A suivante :

$$A = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Le vecteur R des sommes des colonnes est le suivant :

$$R = [2\,4\,2\,2]^T$$

Le vecteur C des sommes des lignes est le suivant :

$$C = [2\,2\,2\,2\,1\,1]^T$$

La phase résolution commence par l'affectation de valeurs à (n - m) variables par l'utilisateur. Chaque fois qu'une variable x_i est connue, l'algorithme essaye de détecter un sous-système d'équations à résoudre simultanément. La matrice A est alors mise à jour. Ce qui correspond à mettre à zéro la $i^{ème}$ colonne de A. On obtient alors une matrice mx(n-1) qu'on notera A^*. La matrice A^* est examinée pour détecter p équations à p variables à résoudre simultanément. En d'autres termes, trouver une matrice carrée pxp notée A_c à extraire de A^*.

Considérons les contraintes représentées par les équations f_i précédentes. Si la résolution commence par la spécification de x_1, la première colonne, correspondant à x_1, est mise à zéro, et on obtient la matrice A^* suivante :

$$A^* = \begin{array}{c} \\ \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{cccccc} & \boxed{1} & & & & \\ & \downarrow & & & & \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array} \begin{array}{l} \\ \leftarrow A_c \\ \\ \\ \end{array}$$

Une matrice A_c (avec p = 1) qu'on peut résoudre est extraite de A^* : la variable x_2 peut être calculée en utilisant l'équation f_1.

De manière générale, une fois qu'une matrice A_c est extraite, on décomposera, si possible, cette matrice en sous matrices A_{c1}, A_{c2} ... avant d'appliquer la phase résolution.

La recherche des matrices Ac est de nature combinatoire, les auteurs proposent une heuristique, qui marche bien selon eux, pour améliorer l'algorithme. Nous proposerons, dans le chapitre 2, une méthode polynomiale, plus efficace, de décomposition et résolution qui ne se base sur aucune heuristique et qui prend en compte tous les schémas de contraintes, qu'ils soient sur- sous- ou bien-contraints.

1.3.2 RESOLUTION SYMBOLIQUE

Nous présentons dans cette section des logiciels de modélisation ayant un solveur symbolique des contraintes. Nous décrivons plus précisément les systèmes développés par Ericson et Yap [EY 88] et Kondo ([Kond 90] et [Kond 92])

• Ericson et Yap [EY 88] ont développé LINETOOL un éditeur géométrique. Ce système est destiné à des chercheurs en géométrie algorithmique, en robotique et en calcul algébrique.

Il permet aux utilisateurs de définir des scènes par la déclaration d'objets géométriques construit à partir de constantes, variables indépendantes, variables dépendantes et contraintes géométriques. Les entités géométriques élémentaires utilisées sont le point, la droite, le segment de droite, le cercle et l'arc de cercle. Les objets géométriques peuvent avoir des motifs de remplissage. Un graphe sauvegardant les liaisons entre les variables dépendantes est construit. Ce graphe doit être acyclique. Chaque contrainte est décrite par une équation polynomiale. Le système résout alors le système de contraintes à l'aide des bases de Gröbner et affiche la scène résultante. Dans le cas où la résolution n'aboutit pas, le système signale à l'utilisateur que son schéma est incohérent i.e le système d'équations n'est pas soluble ou possède une infinité de solutions. L'utilisateur peut, en plus, obtenir des informations concernant les relations spatiales entre différents objets géométriques ; par exemple obtenir, après la résolution, les valeurs des distances entre points non spécifiées au départ.

Cette méthode ne détecte pas les entités géométriques sur- et sous-contraintes. De plus cette méthode est assez lente du fait de l'utilisation des bases de Gröbner lors de la résolution. Enfin, on peut remarquer que c'est à l'utilisateur d'indiquer les relations de dépendances entre les différentes variables pour décomposer le système d'équations en sous-systèmes.

• Kondo ([Kond 90] et [Kond 92]) a développé le logiciel PIGMOD de modélisation géométrique. C'est un outil d'aide à la conception de pièces mécaniques.

Ce système utilise des opérations de modélisation "bien définies" c.à.d chaque opération construit ou définit explicitement l'entité géométrique contrainte (relativement à une autre entité connue). Il sauvegarde, en outre, l'historique des opérations de construction à l'aide d'un graphe pour un maintien futur des contraintes.

Un exemple d'opérations de modélisation est donné dans ce qui suit :

opération-1 : Placer un sommet S1.
opération-2 : Placer un sommet S2 relativement à S1 (la distance a entre ces deux sommets est donnée).
opération-3 : Tracer le segment de droite D1 liant S1 et S2.
opération-4 : Tracer une ligne circulaire A1 de centre S1 et de rayon b donné.
opération-5 : Tracer la droite A2 parallèle à D1 et à distance c donnée de celle ci.
opération-6 : Trouver le sommet S3 en intersectant A1 et A2.
opération-7 : connecter S2 et S3 par le segment de droite D2.
opération-8 : connecter S1 et S3 par le segment de droite D3.

La solution des opérations de modélisation précédentes est donnée par la figure 6. Le graphe sauvegardant l'historique de ces opérations est donné par la figure 7. Si la valeur d'une distance est changée, il y a réexécution des opérations de modélisation affectées par ce changement. Si dans l'exemple précédent la distance c est changée (opération 5) les opérations 6 à 8 doivent être réexécutées car elles dépendent de cette opération.

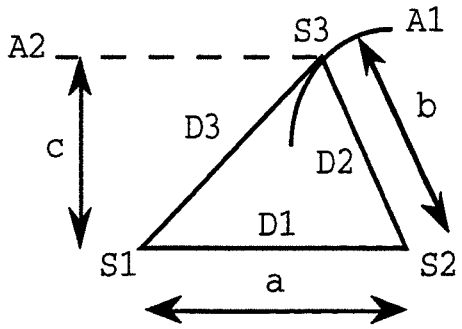


Figure 6.

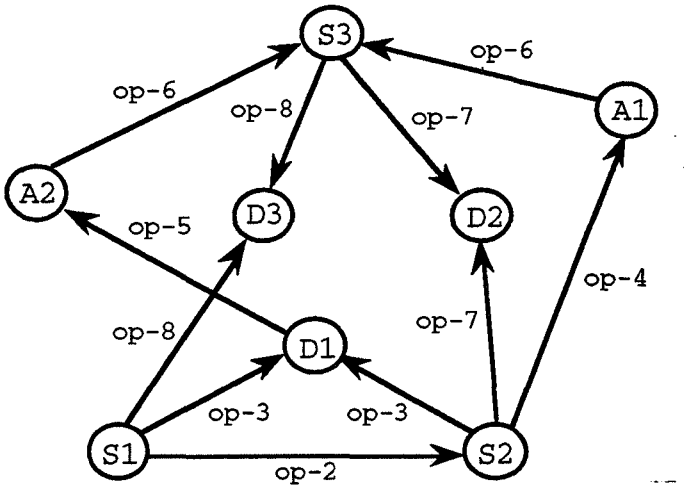


Figure 7.

Le logiciel PIGMOD ne résout que des schémas triangulés en partant de points fixés. Les contraintes utilisées sont de "bas niveau" car l'utilisateur fournit explicitement les procédures à exécuter. De plus, vu la simplicité des problèmes traités, l'utilité des bases de Gröbner n'est pas évidente.

2. APPROCHES GEOMETRIQUES

Ces approches tentent de décomposer le problème des contraintes à résoudre en petits problèmes classiques de géométrie. Ces problèmes sont du type : intersection de courbes (droites, cercles,...), problème d'Appolonius, ... en 2D et intersection de surfaces (plans, sphères,...) en 3D.

Ces approches utilisent soit des règles de déduction et un moteur d'inférence (Aldefeld [Alde 88], Brüderlin ([Brüd 86] et [Brüd 88]), Schreck [Schr 92], Sunde [Sund 87], Lucas, Martin, Martin et Plemenos [LMMP 89], Suzuki, Ando et Kimura [SAK 90], Woodbury [Wood 90], Verroust [Verr 90] et Verroust, Schonek et Roller [VSR 92]), soit des algorithmes opérant sur le graphe de contraintes (Chyz [Chyz 85], Cugini, Devoti et Galli [CDG 85], Cugini, Folini et Vicini [CFV 88], Gaoua, Gardan, Jung et Zakari ([GGJZ 92a] et [GGJZ 92b]) et Owen [Owen 91]).

2.1 METHODES DEDUCTIVES

• Brüderlin ([Brüd 86] et [Brüd 88]) a développé un système de construction d'objets bi- et tri-dimensionnels définis par contraintes. L'algorithme de satisfaction des contraintes est implémenté en Prolog. Les contraintes ainsi que les régles de constructions de cet algorithme sont exprimées de manière symbolique. Dans une première phase, le problème de contraintes est résolu symboliquement. Dans une seconde phase, le système calcule les coordonnés des objets en évaluant les étapes de construction données par la première phase. Cette deuxième phase fait appel à des procédures implémentées en Modula-2.

Les contraintes utilisées sont restreintes aux relations entre points. Elles sont exprimées sous forme de prédicats (ou des faits Prolog). Des exemples de contraintes sont donnés dans ce qui suit :

c(P1, [X, Y])	Les coordonnées du point P1 sont X et Y.
d(P1, P2, d)	d est la distance entre les points P1 et P2.
s(P1, P2, a)	a est la pente de la droite P1P2.
a(P1, P2, P2, b)	b est l'angle P1P2P3.
s(P1, P2) = s(P3, P4)	la droite P1P2 est parallèle à la droite P3P4.

Un exemple de règle de construction est :

```
[c(P1, [X1, Y1]), c(P2, [X2, Y2]), d(P1, P3, R1), d(P2, P3, R2)]
-> [c(P1, [X1, Y1]), c(P2, [X2, Y2]),
    c(P3, intersection(cercle(c(P1), R1), cercle(c(P2), R2)))]
```

Etant donnés les points P1 et P2, la distance entre P1 et P3 et la distance entre P2 et P3, le point P3 est obtenu par l'intersection de deux cercles.

D'après les régles et les contraintes utilisées, le système proposé par Brüderlin ne résout que des schémas de contraintes complètement "triangulés" en partant de points fixés. Le cas 3D n'est explicité qu'à travers des exemples très simples.

• Sunde [Sund 87] propose un logiciel de construction de figures planes. Les concepts d'ensembles CA et CD sont introduits. Deux segment appartiennent à

un même ensemble CA si ils sont contraints en angle. Deux points d'un même ensemble CD sont contraints en distance. La figure solution est obtenue à un déplacement près.

Différents types de régles sont utilisés:

Des règles de création, qui sont déclenchées pour créer des repères lors de la spécification de contraintes :

- Quand une contrainte de distance d entre deux points est donnée, un repère est créé et les coordonnées respectives de ces deux points sont (0,0) et (0,d).
- Quand une contrainte d'angle ß entre deux segments est donnée, un repère angulaire est créé et les directions des deux segments (0 et ß) sont données dans celui-ci.

Des règles de construction, qui permettent de guider le "calcul" de la figure solution en réunissant les repères qui doivent l'être :

- Si l'angle entre deux segments est connu et que chaque segment est fixé dans un ensemble CA, ces deux ensembles sont alors réunis.
- Si deux ensembles CD ont un point commun et sont contraints en angle, ces deux ensembles sont réunis.

Si tous les points appartiennent à un même ensemble CD, le schéma de contraintes est résolu. Cette méthode permet aussi la détection des entités sur-contraintes.

Sunde ne donne que quelques régles du système développé et son approche ne prend en compte que les entités géométriques point et droite. Les cercles de rayon inconnu ne sont pas considérés.

• Aldefeld [Alde 88] procéde, comme Brüderlin, à la résolution du problème de contraintes en deux phases. Une première phase symbolique utilise des règles et un moteur d'inférences. Une seconde phase numérique évalue le plan de construction établi par la première phase.

Les types de contraintes sont comparables. Des exemples de ces contraintes sont donnés par ce qui suit :

(Distance P_1 P_2 d)	d est la distance entre les points P_1 et P_2,
(Coords P_0 (x_0, y_0))	Les coordonnées de P_0 sont (x_0, y_0),
(Parallèle D_1 D_2)	La droite D_1 est parallèle à la droite D_2,
(Centre P C)	P est le centre du cercle C.

L'historique des inférences est sauvegardé. Il correspond aux règles dont l'application a abouti. Quand une contrainte est supprimée, le système recherche et supprime dans l'historique des inférences les faits déduits de cette contrainte. Cet historique fournit aussi des explications quand le système détecte des entités sur-contraintes. Aldefeld utilise deux types de contraintes pour choisir la solution dans le cas de solutions multiples. Elles sont du type "point à droite ou à gauche d'une ligne" ou "point précéde ou suit un point le long d'une ligne".

Cette méthode est assez lente et ses limites sont du même type que celles de l'approche de Brüderlin.

• Braun ([Brau 88] et [Brau 89]) a developpé GEOPHILE un outil de programmation de constructions géométriques. Les contraintes utilisées sont du type : point origine d'un segment, droite passant par un point,..., point milieu d'un segment. Chaque relation de la construction est décrite par une équation de la forme : X = f(Y,Z,...), f étant une fonction dont on connait une description procédurale. On ne détermine un élément que lorsque tous les éléments dont il dépend sont déterminés. Cet ordre de dépendance entre les objets d'une construction est un ordre partiel représenté sous forme d'un graphe orienté sans cycles.

• Suzuki, Ando et Kimura [SAK 90] ont développé un logiciel de dessins de pièces mécaniques. Les contraintes et les règles de constructions de cet algorithme sont exprimées de manière symbolique. Elles sont similaires à celles utilisées par Brüderlin. Les auteurs introduisent, en plus, l'entité géométrique cercle et les contraintes correspondantes. Les dépendances entre les entités géométriques sont sauvegardées. Elles permettent de détecter les entités géométriques sur- et sous-contraintes.

Les limites de ce logiciel sont du même type que celles de l'approche de Brüderlin.

• Schreck [Schr 92] a développé PROGE, un système résolvant des exercices de construction en géométrie élémentaire plane. Schreck a pris comme point de départ les travaux de Buthion [Buth 79]. Ce système, à partir d'un énoncé déclaratif d'un exercice de lycée ou de collège, engendre un programme procédural de construction. PROGE fonctionne en deux phases. Dans la première, il résout formellement un problème de construction à la règle et au compas à l'aide d'une figure appelée figure formelle. Dans la seconde, il permet d'interpréter numériquement le programme de construction obtenu et de tracer les dessins correspondants. Il donne, en outre, des explications du raisonnement fait.

Dans la base de règles couramment utilisée par PROGE, on a la règle typique suivante :

1 # **si** [distance (A, B) = d] et [connu A, connu d, pas_connu B]
alors [B est_sur ccr(A,k)].

Cette règle exprime que si la distance d entre deux points A et B est connue, que le point A est connu, et que le point B n'est pas encore construit, alors B appartient au cercle de centre A et de rayon la distance d.

PROGE résout des constructions comprenant des cercles de rayon inconnu. Cependant le nombre de règles utilisées est grand et l'une des conditions de réussite est que l'énoncé soit correctement posé. Les entités géométriques sur- et sous-contraintes ne sont pas détéctées.

• Verroust ([Verr 90] et [VSR 92]) a pris comme point de départ l'approche de Sunde [Sund 87]. Elle utilise des règles implantées dans un système expert pour construire une figure polygonale dont certains sommets ou segments sont contraints par des relations de distances ou d'angles. Les régles permettent aussi de détecter des cas de sur-contrainte d'une partie du dessin.

Différents types de régles sont utilisés. Les principales régles de construction sont les suivantes:

Les **règles du triangle** permettent de construire un triangle connu par la donnée de ses trois distances, de deux distances et un angle ou une distance et deux angles.

Exemple :

Soit le triangle ABC (figure 8) connu par ses trois distances. Deux points de ce triangle (A et B) sont fixés, le troisième point (C) est obtenu de l'intersection de deux cercles.

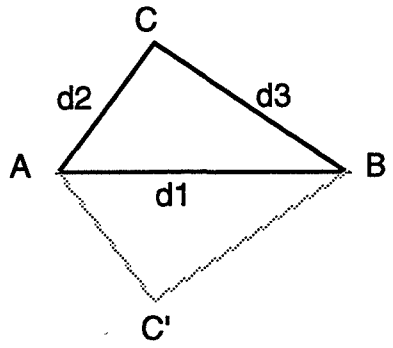


Figure 8.

les **règles du quadrilatère** permettent de construire un quadrilatère connu par la donnée de quatre distances et un angle, trois distances et deux angles ou deux distances et trois angles entre ses différents sommets et segments.

La **règle du parallélogramme** concerne des cas particuliers de quadrilatère et les polygones ayant plus de quatre sommets. Cette régle permet de réunir deux segments contraints en angle mais non adjacents.

L'ensemble de la figure est fixé quand il existe un repère pour lequel les coordonnées de tous les points sont connus.

Définitions :

- Un schéma est bien-contraint si il existe un nombre suffisant de contraintes permettant de le déterminer sans ambiguités.

• Un schéma de contraintes est dit simple si le graphe associé peut être couvert par un cycle simple.

Un schéma simple bien contraint constitué de n points peut avoir (n-1) contraintes d'angles et (n-2) contraintes de distances, (n-2) contraintes d'angles et (n-1) contraintes de distances ou (n-3) contraintes d'angles et n contraintes de distances.

Verroust montre que l'ensemble des régles de construction permet le calcul d'une solution à un déplacement près pour les schémas simples.

Exemple :

Soit un schéma simple contraint par n-1 angles et n-2 distances. Toutes les directions sont fixées entre elles. Elles appartiennent au même ensemble CA. Supposons que les contraintes de distances concernent deux ensembles connexes de points (Figure 9).

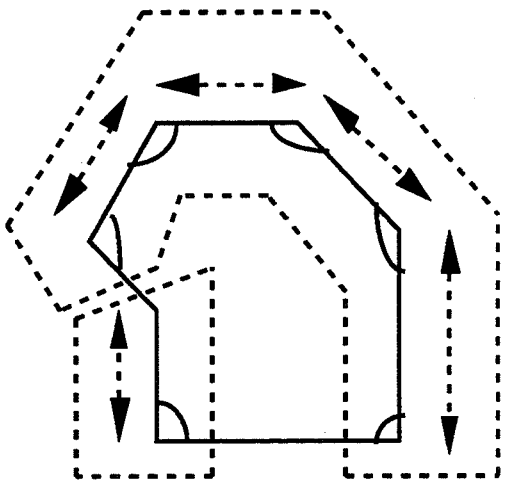


Figure 9.

Les n points sont répartis en deux ensembles CD. Une des régles du quadrilatère permet la réunion de ces deux ensembles CD.

Les régles utilisées par Verroust pour vérifier les cas de sur-contraintes sont les suivantes :

- S'il existe une contrainte excédentaire d'angle entre deux segments déjà fixés dans un même repère angulaire, il y a sur-contrainte angulaire pour ces deux segments.
- S'il existe une contrainte excédentaire de distance entre deux points déjà fixés dans un repère, il y a sur-contrainte de distance entre ces deux points.

Comme pour la méthode de Sunde [Sund 87], Verroust ne prend en compte que les entités géométriques point et droite. Les schémas de contraintes comprenant des cercles de rayon inconnu ne sont pas résolus.

2.2 AUTRES METHODES

• Cugini, Folini et Vicini [CFV 88] ont développé un logiciel de CAO. Les constructions sont modelisées par un graphe orienté sans cycles. Les sommets de ce graphe représentent les entités géométriques (points, droites et cercles) et les arcs les contraintes entre ces entités. La chronologie de la construction est sauvegardée dans une liste contenant les entités géométriques. A chaque modification d'une contrainte, il y a recalcul de toutes les entités géométriques de la sous-liste commençant à la première entité affectée. Ceci peut mener à des calculs inutiles que nous verrons dans ce qui suit. Un exemple de construction est donné par la figure 14. Le graphe correspondant est donné par la figure 15. Les arcs de ce graphe représentent des relations de distance entre les points A, B, C, D, E et F.

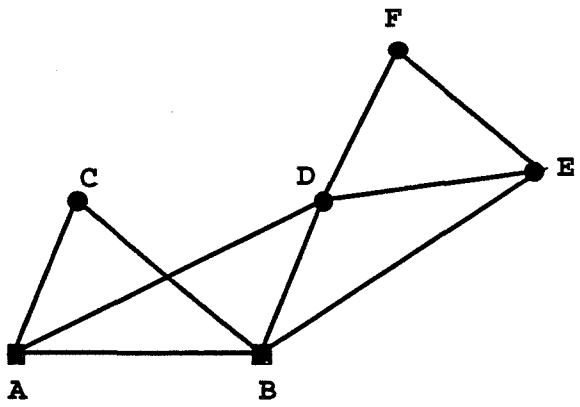


Figure 14.

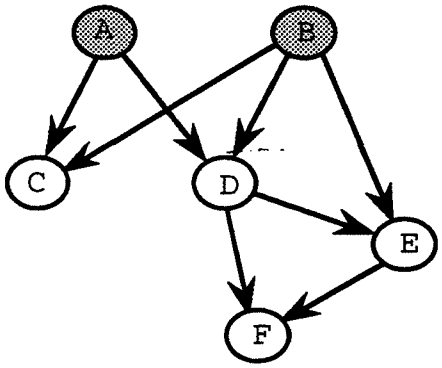


Figure 15.

A partir des points A et B, fixés au départ, on construit les points C, D, E et F ; la liste sauvegardant l'historique est la suivante (C D E F). Si, par exemple, nous modifions la contrainte de dimension entre les points A et C, toutes les entités de cette liste seront recalculées alors qu'en réalité nous n'avons besoin que de la modification du point C. Cette méthode serait plus pénalisante encore si la "branche" (D E F) était plus importante. Pour ne modifier que les entités géométriques qui doivent l'être, on procéde comme [Ait 92] en faisant un parcours du graphe à partir de la contrainte affectée jusqu'à arriver à une feuille de ce graphe. Les sommets traversés doivent être modifiés selon un ordre donné par le tri topologique sur ces sommets.

• Owen [Owen 91] étudie les configurations générales de contraintes d'angle et de distance entre points, lignes et cercles sur un plan (le rayon des cercles est connu). Les valeurs des angles et des distances sont connues. La méthode proposée par Owen résout une classe de problèmes similaire à celle résolue par la méthode de Verroust.

L'objectif des travaux d'Owen est d'étudier les équations obtenues lors de la définition des entités géométriques par des contraintes d'angles et de distances relatives. Spécifiquement, Owen utilise la théorie de Galois des équations pour démontrer l'équivalence des trois propriètés suivantes :

> ***1.*** *Une figure géométrique plane est constructible à la régle et au compas.*
>
> ***2.*** *Les coordonnées des éléments de la figure peuvent être calculées algébriquement en utilisant uniquement des opérations arithmétiques plus la racine carrée.*
>
> ***3.*** *Ces coordonnées appartiennent à un corps qui est une extension algébrique, de degré 2^n, sur les données initiales.*

D'après Owen, l'algorithme qu'il propose permet de déduire les coordonnées de la figure à partir des valeurs des dimensions .

Remarque :

En fait, contrairement à ce qu'Owen affirme et prétend prouver, son algorithme ne résout pas tous les schémas de contraintes constructibles à la règle et au compas. Un exemple d'un tel cas sera donné plus loin.

Un graphe non orienté est utilisé. Les sommets de ce graphe représentent les entités géométriques et les arêtes représentent les contraintes entre les entités. Un exemple de graphe de contraintes est donné par la figure 10.

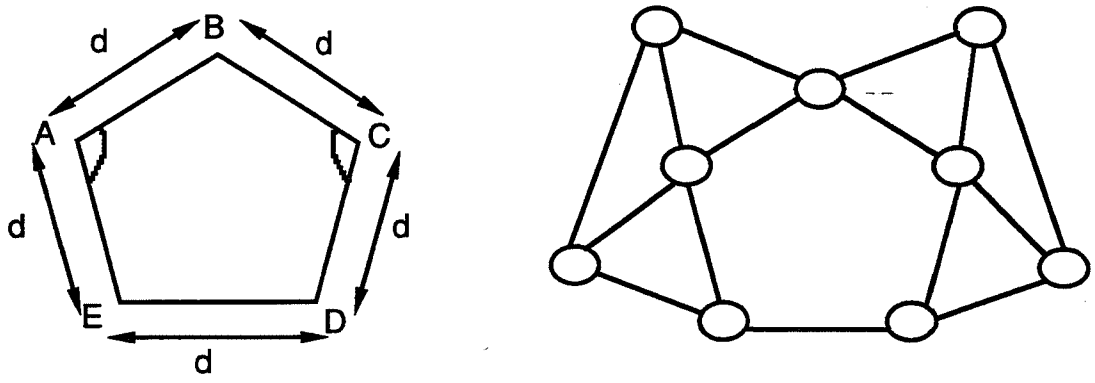


Figure 10 : Une figure et son graphe de contraintes.

Rappels :

- Un graphe est connexe si pour tout couple de sommets, il existe une chaîne (séquence d'arêtes) reliant ces deux sommets.
- Un sommet d'articulation d'un graphe connexe est un sommet tel que le sous-graphe obtenu après la suppression de ce sommet ne soit plus connexe.
- Un graphe, ayant plus de deux sommets, sans sommet d'articulation est biconnexe.

Algorithme :

La déduction des entités géométriques s'effectue en deux phases.

Phase d'analyse.

1. Diviser le graphe des contraintes en composantes biconnexes.
Un graphe bien contraint contient une seule composante biconnexe.

2. Diviser chaque composante biconnexe en sous-composantes en insérant des arêtes virtuelles si nécessaire entre deux sommets d'articulation. Une sous-composante peut être une arête, un triangle, un polygone ou un autre. Un graphe bien contraint ne contient pas de polygones.

3. A chaque paire d'articulation sans arête simple et avec exactement un sous-graphe, enlever les arêtes virtuelles. On enlève les arêtes virtuelles pour permettre d'autres décompositions.

4. L'algorithme s'arrête quand on ne peut plus diviser le graphe des contraintes. Les équations ne peuvent pas être résolues quadratiquement (i.e en utilisant les opérations arithmétiques plus la racine carrée), selon l'auteur, si on obtient un sous-graphe tri-connexe.

Phase de calcul.

Cette phase procéde dans l'ordre inverse de la phase précédente. Premièrement les triangles sans arêtes virtuelles sont "calculés" et replacés pour former les sous-graphes. Les valeurs associées aux arêtes virtuelles sont obtenues à partir de ces sous-graphes. La procédure est réitérée jusqu'à déterminer complètement la figure.

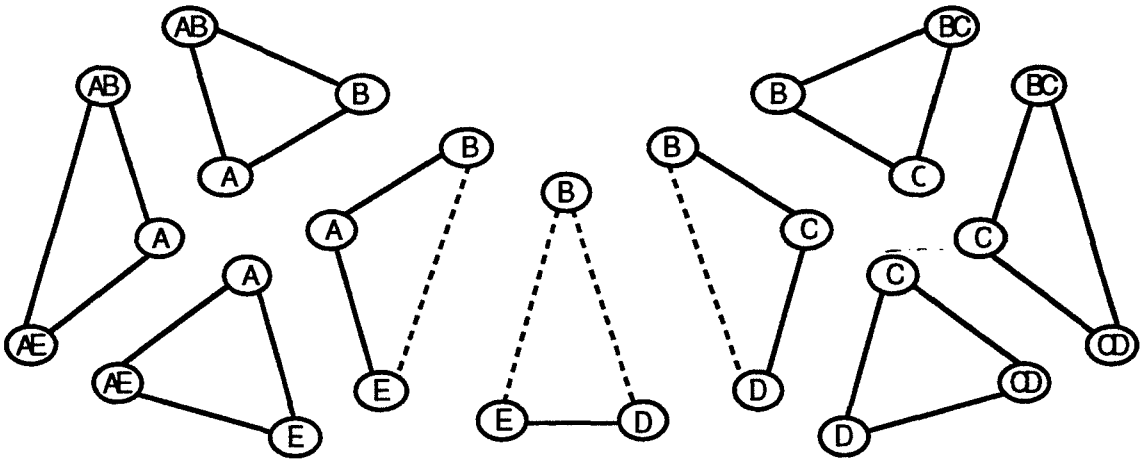
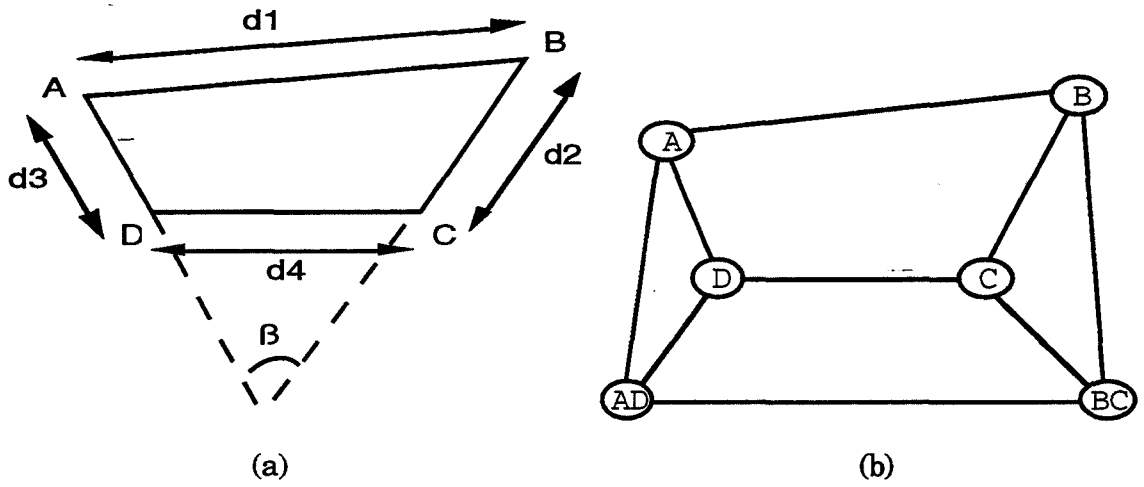


Figure 11 : Décomposition du graphe des contraintes.

La décomposition récursive du graphe des contraintes de la figure 10 avec insertion d'arêtes virtuelles est donnée par la figure 11. Cette décomposition ne comporte que des sous composantes triangulées. On pourra alors "résoudre" ces triangles et propager les résultats obtenus.



(a) (b)

Figure 12.

L'algorithme d'Owen ne résout pas tous les schémas de contraintes constructibles à la règle et au compas. Soit le quadrilatère ABCD donné par ses quatre distances et un angle entre deux cotés opposés AD et BC (figure 12a). Le graphe des contraintes correspondant est donné par la figure 12b.

La figure 12a est constructible à la règle et au compas mais son graphe de contraintes est triconnexe, donc ce cas n'est pas résolu par l'algorithme d'Owen.

Pour construire la figure 12a à l'aide de la régle et du compas (figure 13) on procéde de la façon suivante : on fixe les points A et D, on trace le segment AE tel que l'angle (AE,AD) soit égal à ß (l'angle donné entre AD et BC) et la distance AE soit égale à la distance CB. Le segment AE est parallèle au segment CB. Le point C est alors obtenu par l'intersection des deux cercles de centres D et E et de rayons respectifs d4 et d1. Le point B est enfin obtenu en intersectant les deux cercles de centres C et A et de rayons respectifs d2 et d1.

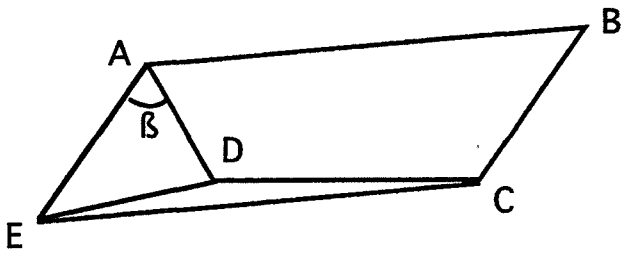


Figure 13.

Owen n'indique pas non plus comment le choix de la solution est effectué dans le cas de solutions multiples.

B. REALISABILITE.

Il s'agit de calculer les coordonnées de sommets dans un espace (généralement le plan ou l'espace tridimensionel) à partir des distances entre ces sommets. Le problème est formalisé par la donnée d'un graphe où les arêtes représentent les distances connues entre les sommets.

Le problème de la réalisabilité a été étudié en théorie des graphes, entre autres, par Laman [Lama 70], Yemini [Yemi 79], Lovasz et Yemini [LY 82], Recski [Recs 85] et Hendrickson [Hend 90].

Une réalisation du graphe est l'affectation de coordonnées à chaque sommet du graphe telles que les contraintes de distance soient satisfaites. Ce problème a été montré comme étant NP-complet [Saxe 79]. Un graphe est dit rigide si il n'admet pas de déformation continue (autre que le déplacement de toute la structure). Le problème de la réalisabilité peut être étendu pour prendre en compte des sommets de type droite et où les arêtes peuvent représenter des relations d'incidences ou d'angles. En fait toutes les contraintes invariantes par translation et rotation peuvent être prises en compte.

Ce problème concerne, outre la théorie des graphes, le calcul des structures (ou tout autre système barres-assemblages) en résistance des matériaux [Lama 70] et la chimie (détermination des différentes configurations de molécules) [Hend 90].

1. INTRODUCTION.

Soit le G = (V,E,l) un graphe valué ayant n sommets et m arêtes. Un nombre réel positif est associé à chaque arête. Le problème de la réalisation de graphe est de calculer les coordonnées de chaque sommet telles que la distance entre deux sommets adjacents (reliés par une arête) soit égale au nombre associé à cette arête.

Le problème lié à la réalisation de graphe est de déterminer si le nombre de réalisations est fini c.à.d si il existe un nombre fini de solutions modulo des translations, des rotations ou des symétries de toute la figure. Dans ce cas le graphe est dit rigide. Certains graphes peuvent avoir une infinité de réalisations. L'exemple d'un tel graphe est donné par la figure 16.

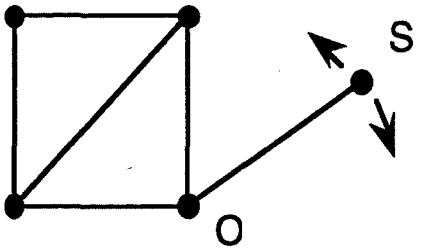


Figure 16.

Le sommet S du graphe précédent peut pivoter librement autour de O.

Un graphe G est dit flexible (non rigide ou sous contraint) s'il peut subir des déformations continues (autre que les déplacements de toute la structure) tout en satisfaisant toutes les contraintes.

Une réalisation du graphe G est une fonction f liant les sommets du graphe G aux points de l'espace euclidien. La combinaison d'un graphe et d'une réalisation est appelée structure, et est notée F.

2. RIGIDITE DES STRUCTURES.

2.1. RAPPELS

Cette section rappelle la théorie de la rigidité des structures tridimensionnelles ou bidimensionnelles. Etant donnée la réalisation d'un graphe, cette théorie teste la rigidité de la structure obtenue.

Soient G = (V,E) un graphe ayant n sommets et m arêtes et $P_1, P_2, \ldots, P_n$ des points de coordonnées (x_1,y_1,z_1), (x_2,y_2,z_2), ... , (x_n,y_n,z_n) (dans le cas d'un espace à trois dimensions). Une arête entre les points P_i, P_j indique que la distance entre ces deux points est une constante C :

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} = C = \sqrt{c_{ij}}. \qquad (1)$$

Supposons que la structure soit en mouvement (même si cette structure est rigide, elle peut subir une translation par exemple). Les coordonnées des points P_i sont alors des fonctions du temps $(x_i(t), y_i(t), z_i(t))$.

En élevant l'équation (1) au carré on obtient pour chaque arête E_{ij}:

$$(x_i(t) - x_j(t))^2 + (y_i(t) - y_j(t))^2 + (z_i(t) - z_j(t))^2 = c_{ij} . \qquad (2)$$

En différenciant ces équations et en divisant par deux on obtient :

$$(x_i(t) - x_j(t)).(\dot{x}_i(t) - \dot{x}_j(t)) + (y_i(t) - y_j(t)).(\dot{y}_i(t) - \dot{y}_j(t)) + (z_i(t) - z_j(t)).(\dot{z}_i(t) - \dot{z}_j(t)) = 0 . \qquad (3)$$

où $\dot{x}_i(t)$ est $\frac{\partial}{\partial t} x_i(t)$.

Si (x_i,y_i,z_i) et $(\dot{x}_i,\dot{y}_i,\dot{z}_i)$ sont notés par p_i et v_i (position et vitesse) on a :

$$(v_i - v_j).(p_i - p_j) = 0 \quad \text{pour chaque arête } E_{ij}. \qquad (4)$$

i.e la différence des vecteurs positions est perpendiculaire à la différence des vecteurs vitesses.

L'équation (3) peut se réécrire comme suit:

$$(x_i(t) - x_j(t)).\dot{x}_i(t) + (x_j(t) - x_i(t)).\dot{x}_j(t) + \cdots + (z_i(t) - z_j(t)).\dot{z}_i(t) + (z_j(t) - z_i(t)).\dot{z}_j(t) = 0 . \qquad (5)$$

Les m équations relatives aux m arêtes sont traditionnellement exprimées par le système suivant :

$$A.u = 0 \qquad (6)$$

où

$$u = (\dot{x}_1, \dot{x}_2, \ldots, \dot{x}_n, \dot{y}_1, \dot{y}_2, \ldots, \dot{y}_n, \dot{z}_1, \dot{z}_2, \ldots, \dot{z}_n)$$

$A = (a_{kl})$ est une matrice $m_x 3n$

$$a_{kl} = \left\{ \begin{array}{ll} x_i - x_j & \text{si l'arête k joint les sommets i et j, et } l = i \\ y_i - y_j & \text{si l'arête k joint les sommets i et j, et } l = i + n \\ z_i - z_j & \text{si l'arête k joint les sommets i et j, et } l = i + 2n \\ 0 & \text{autrement} \end{array} \right\}$$

par exemple pour la figure 17 on a la matrice A suivante :

$$A = \left\langle \begin{array}{ccccccccc} x_1 - x_2 & x_2 - x_1 & 0 & y_1 - y_2 & y_2 - y_1 & 0 & z_1 - z_2 & z_2 - z_1 & 0 \\ x_1 - x_3 & 0 & x_3 - x_1 & y_1 - y_3 & 0 & y_3 - y_1 & z_1 - z_3 & 0 & z_3 - z_1 \\ 0 & x_2 - x_3 & x_3 - x_2 & 0 & y_2 - y_3 & y_3 - y_2 & 0 & z_2 - z_3 & z_3 - z_2 \end{array} \right\rangle$$

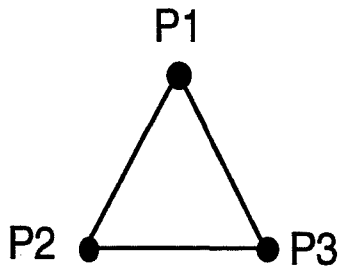


Figure 17.

L'équation (6) a une solution évidente u = 0. Cependant ceci ne signifie pas que la structure est forcément rigide. Par exemple, si on considère la translation de toute la structure rigide par un vecteur (a,b,c), on peut définir un vecteur u :

$$u = \{ \dot{x}_i = a, \dot{y}_i = b, \dot{z}_i = c \quad \text{pour } i = 1,2, \ldots,n \}$$

tel que

$$A.u = 0$$

Plus généralement l'équation (6) a des solutions non triviales si r(A) < 3n, où r(A) désigne le rang de la matrice A. Les solutions non triviales provenant des translations et rotations de tout l'espace tridimensionnel forment un espace

vectoriel de dimension six. Donc, pour avoir d'autres solutions, il faut que r(A) < 3n - 6. (on supposera par la suite que n ≥ 3, i.e il y a au moins deux arêtes dans la structure).

Dans le cas bidimensionnel, l'équation (6) aura des solutions non triviales si r(A) < 2n. Les solutions non triviales provenant des translations et rotations de l'espace bidimensionnel forment un espace vectoriel de dimension trois (il suffit d'appliquer deux translations et une rotation à une structure 2D pour l'amener à une position déterminée). Donc pour que l'équation (6) ait d'autres solutions il faut que r(A) < 2n - 3 (on supposera par la suite que n ≥ 2).

Lemme [Recs 86] :

Soit une structure 3D notée F ayant au moins trois sommets et vérifiant r(A) < 3n - 6 alors cette structure est non rigide.

Une structure tridimensionnelle est rigide si le rang de sa matrice A, r(A) = 3n - 6. Si une structure est non rigide les solutions de l'équation (6) déterminent les parties flexibles de cette structure.

De même, dans le cas 2D, une structure F ayant au moins deux sommets et telle que le rang de sa matrice A, r(A) < 2n - 3 est non rigide. Cette structure est rigide si r(A) = 2n - 3.

Le test de rigidité (utilisé en ingénierie) de structures se fait par un algorithme polynomial : le calcul de rangs de matrices.

Exemples :

Test de rigidité des deux structures planes de la figure 18 en construisant leur matrice A.

La matrice A de la figure 18.a (avec n = 3 et m = 3) est la suivante :

$$A = \begin{pmatrix} 1 & -1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & -1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Le rang de cette matrice r(A) = 3, cette figure est donc rigide car r(A) = 2n - 3.

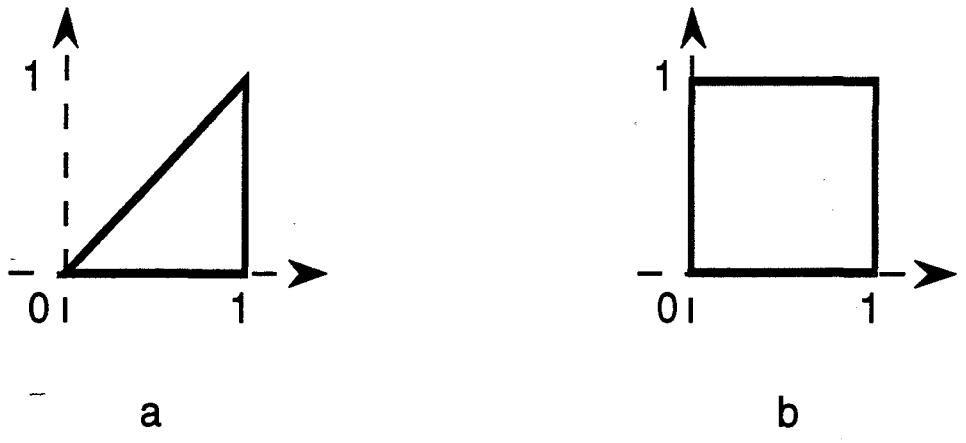


Figure 18.

La matrice A de la figure 18.b (avec n = 4 et m = 4) est la suivante :

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \\ -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

r(A) = 4 < 5 = 2n - 3, donc la figure 18.b est non rigide. La figure 19 donne une déformation possible de la figure 18.b.

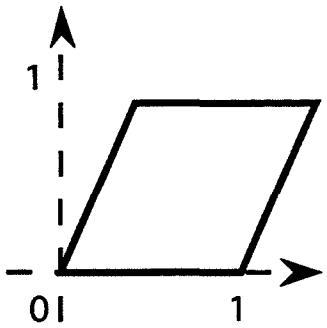


Figure 19.

Soient les deux corollaires [Recs 86] suivants :

Corollaire :

Si une structure tridimensionnelle ayant n sommets et m arêtes est rigide alors $m \geq 3n - 6$.

Corollaire :

Si une structure bidimensionnelle ayant n sommets et m arêtes est rigide alors $m \geq 2n - 3$.

Ces conditions sont nécessaires mais non suffisantes.

Cependant, si seulement le graphe d'une structure F est connu (on n'a pas de réalisation), la matrice A ne peut être calculée et r(A) non plus. Pour contourner ce problème, on ne considérera dans ce qui suit que les structures génériques i.e, mathématiquement parlant les coordonnées des sommets du graphe associé sont algébriquement indépendantes sur l'ensemble des rationnels. Cette condition est trop forte et irréaliste ; en pratique il suffit d'éviter certaines configurations. Le graphe illustré par la figure 20 est un exemple d'un tel cas. Les étiquettes des arêtes sont les valeurs des distances entre les sommets. Ce graphe est génériquement rigide mais réellement non rigide.

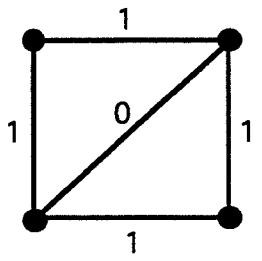


Figure 20.

Le test de rigidité se fera uniquement par l'étude du graphe associé à la structure (les valeurs des distances associées aux arêtes sont ignorées). On présentera dans ce qui suit, pour le cas bidimensionnel, une condition nécessaire et suffisante pour tester la rigidité de telles structures en termes de graphes.

2.2.RIGIDITE DES STRUCTURES GENERIQUES EN 2D :

Pour tester la rigidité de telles structures planes, la condition $m \geq 2n - 3$ n'est pas suffisante. Une structure ayant exactement n sommets et 2n -3 arêtes peut

être non rigide (figure 21), puisque il peut y avoir un "sous système" ayant n' sommets et m' arêtes satisfaisant $m' > 2n' - 3$ et donc la partie restante peut être flexible.

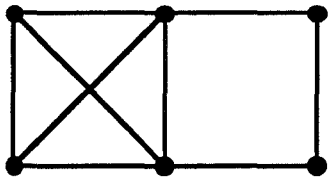


Figure 21.

Le théorème de Laman [LAMA 70] donne une condition nécessaire et suffisante pour la rigidité des structures génériques.

Théorème [Lama 70] :

> *Si une structure 2D R (non forcément planaire) ayant n sommets et* $m = 2n - 3$ *arêtes est générique, R est rigide* si et seulement si $m = 2n - 3$ *et* $m' \leq 2n' - 3$ *pour chaque sous-graphe de R ayant n' sommets et m' arêtes.*

Nous présentons ultérieurement deux algorithmes de test effectif de la rigidité des structures 2D.

2.3. RIGIDITE DES STRUCTURES GENERIQUES EN 3D :

La généralisation du théorème de Laman au cas tridimensionnel est fausse. La figure 22 montre une structure tridimensionnelle ayant $n = 8$ sommets et $m = 18$ arêtes. Cette structure vérifie la condition $m = 3n - 6$ et la condition $m' \leq 3n' - 6$ est vraie pour chaque sous système ayant m' arêtes et n' sommets. Cependant cette structure est flexible. Ses deux parties (gauche et droite) peuvent subir des rotations autour de l'axe passant par le plus haut et le plus bas sommet.

Pour des structures de dimension supérieure à deux, il n'existe pas de caractérisation de la rigidité en termes de graphes. Une première solution pour contourner ce problème est de calculer symboliquement le rang de la matrice de rigidité. Cependant le déterminant de cette matrice peut avoir un nombre exponentiel de termes. Cette méthode nécessite donc un temps exponentiel. Une deuxième solution est basée sur le théorème de Gluck [Gluc 75] suivant :

Théorème :

Si un graphe admet une réalisation rigide, alors toute les réalisations génériques sont rigides.

D'après Hendrickson [Hend 92], il suffit de sélectionner une réalisation aléatoire et de calculer ensuite le rang de la matrice de rigidité, pour avoir la réponse correcte avec une probabilité de un.

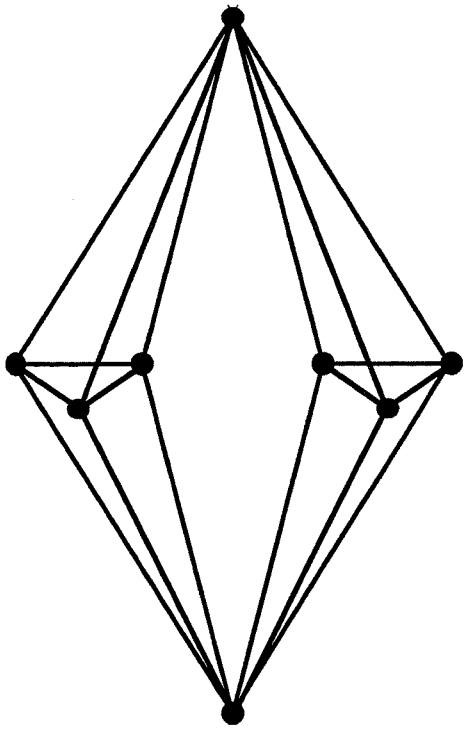


Figure 22.

3. TEST DE RIGIDITE.

Une utilisation naïve du théorème de Laman énoncé précédemment nécessite le test d'un nombre exponentiel de sous systèmes : ce théorème ne peut être exploité pratiquement sous sa forme initiale. Nous présentons dans ce qui suit deux algorithmes plus efficaces qui testent la rigidité des structures 2D.

3.1 ALGORITHME DE LOVASZ ET YEMINI

Lovasz et Yemini [LY 82] proposent une autre formulation de la condition de Laman pour tester la rigidité des graphes bidimensionnels.

Théorème [LY 82] :

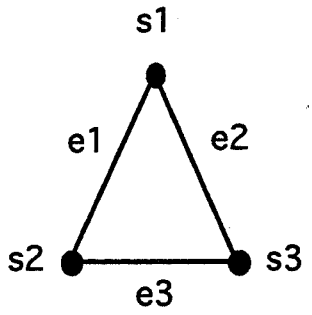
> *Soit R une structure générique 2D (non forcément planaire) ayant n sommets et m = 2n - 3 arêtes et G représentant son graphe . R est rigide si et seulement si l'ensemble des arêtes du graphe G_x obtenu à partir du graphe G en dédoublant une arête x de G peut être parcouru par deux arbres recouvrants disjoints pour chaque arête x.*

3.2 ALGORITHME DE HENDRICKSON

On exposera brièvement l'algorithme en $O(n^2)$ de Hendrickson [Hend 92] basé sur les couplages de graphes bipartis.

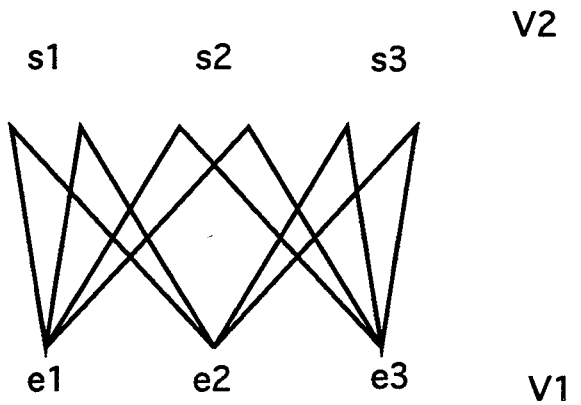
On introduit la notion de graphe biparti B(G) qu'on génère à partir du graphe initial G = (V,E). Le premier ensemble V1 de sommets du graphe B(G) est donné par les arêtes de G : les sommets de V1 représentent les contraintes. Le second ensemble V2 de sommets de B(G) est composé de deux copies de chaque sommet de G : chaque sommet représente en fait une coordonnée du sommet de G. Chaque sommet "arête" appartenant à V1 est relié aux deux copies de V2 de chaque sommet incidents à cette arête. Le graphe B(G) a donc 2n + m sommets et 4m arêtes où m et n sont respectivement le nombre d'arêtes et le nombre de sommets du graphe G.

Un exemple de correspondance entre G et B(G) est donné par les figures 23 et 24 :



G

Figure 23.



B(G)

Figure 24.

Le graphe biparti ainsi défini permet d'exprimer la condition de Laman sous une autre forme, donnée par le théorème suivant :

Définition :

Les arêtes d'un graphe G = (V, E) sont indépendantes dans l'espace bidimensionnel si et seulement si aucun sous-graphe G' = (V', E') ne posséde plus de 2n' - 3 arêtes.

Théorème :

Soit le graphe G = (V,E), les quatre propriétés suivantes sont équivalentes :

(1) Les arêtes de G sont indépendantes dans l'espace plan .

(2) Pour chaque arête (a,b) dans G, le graphe G_{ab} formé en ajoutant trois arêtes supplémentaires entre a et b n'a pas de sous graphe pour lequel m' > 2n'.

(3) Pour chaque arête (a,b), le graphe biparti $B(G_{ab})$, issu de G_{ab}, n'a pas de sous ensemble de V1 qui soit adjacent à un plus petit sous ensemble de V2.

(4) Pour chaque arête (a,b), le graphe biparti $B(G_{ab})$, issu de G_{ab}, a un couplage parfait (complet) de V1 à V2.

L'algorithme de Hendrickson est basé sur la partie (4) du théorème précédent. L'idée principale est d'augmenter l'ensemble des arêtes indépendantes à chaque étape. Une arête est rajoutée à une base d'arêtes si elle est indépendante des autres arêtes de cette base. Si 2n - 3 arêtes indépendantes sont trouvées alors le graphe est rigide.

Soit E' un ensemble d'arêtes indépendantes. Cet ensemble d'arêtes et les sommets de G forment le graphe G'. On cherche à déterminer si une autre arête e est indépendante de E'. Le graphe G" est obtenu par l'ajout de e dans G'. Cette arête e est indépendante de E' si et seulement si il existe un couplage parfait (complet) du graphe B(G") après que l'arête e soit quadruplée dans G".

Pour améliorer son algorithme Hendrickson évite certains tests inutiles. Définissons un sous graphe de Laman comme un sous-graphe ayant n' sommets et 2n' - 3 arêtes indépendantes. Le couplage n'aboutira pas si la nouvelle arête à tester appartient à un sous graphe qui a déjà 2n' - 3 arêtes indépendantes, aucune autre arête ne pouvant y être insérée. Il est donc inutile de faire les calculs et tests concernant cette arête.

Algorithme :

Base := ∅;
Pour chaque sommet v
    Marquer chaque sommet dans les sous-graphes de Laman comprenant v et effacer le marquage des autres sommets;

Pour chaque arête (u,v)
Si u n'est pas marqué **Alors**
Si (u,v) est indépendant de Base **Alors**
rajouter (u,v) à Base;
créer un sous-graphe de Laman composé de (u,v);
Sinon (* Un sous-graphe de Laman a été identifié *)
Union des sous-graphes de Laman ayant une arête en commun;
Marquer chaque sommet dans les sous-graphes de Laman comprenant v;

C. CONCLUSION

Il n'existe pas de méthode universelle de résolution d'un problème de contraintes. Le choix d'une méthode de résolution dépend du domaine d'application.

Par exemple, pour définir une figure constructible à la règle et au compas on optera pour une méthode géométrique. Dans un cas plus général on choisira une méthode algébrique. Si le temps de calcul n'est pas crucial on résoudra symboliquement le système de contraintes.

Chaque méthode a ses avantages et ses inconvénients :

• Les méthodes algébriques numériques ne sont pas toujours convergentes. De plus, s'il y a convergence, le résultat obtenu peut ne pas être la solution voulue car celui-ci dépend du choix des valeurs initiales. Un autre inconvénient est le nombre d'équations qui est fonction de la complexité de la figure. Il est souvent nécessaire d'utiliser des algorithmes de décomposition de systèmes d'équations. Un autre grief contre ces méthodes est l'absence d'explication géométrique dans le cas des sur et sous-contraintes.

Un avantage des méthodes numériques est qu'elles permettent de traiter un grand nombre de types de contraintes. On peut en outre facilement ajouter de nouveaux types de contraintes .

Ajoutons, pour terminer, que les méthodes de résolution, numériques ou symboliques, des systèmes d'équations algébriques sont un domaine de

recherche très actif actuellement (comme le prouve par exemple la tenue du congrès SEA'93 à Marseille, entièrement consacré à ce thème).

• Les méthodes géométriques ne résolvent pas tous les systèmes de contraintes. Un autre problème de ces méthodes est le nombre de cas à traiter qui augmente fortement si on a un très grand nombre de types de contraintes géométriques. De plus, l'ajout d'un nouveau type de contraintes n'est pas simple.

Par contre, pour une classe de problèmes restreinte, ces méthodes peuvent être plus efficaces que les méthodes algébriques. L'utilisateur peut, en outre, avoir les détails du processus de résolution.

CHAPITRE 2

REDUCTION DE SYSTEMES DE CONTRAINTES

La modélisation géométrique par contraintes aboutit à de grands systèmes d'équations. Ce chapitre étudie les graphes bipartis sous-jacents à ces systèmes d'équations. Nous montrons qu'il est possible de décomposer polynomialement ces systèmes en sous-systèmes bien contraints, sur-contraints et sous-contraints à partir du graphe biparti. Nous verrons aussi une méthode efficace de décomposition des systèmes bien contraints en sous-systèmes irréductibles. Ces décompositions accélérent grandement la résolution dans le cas des grands systèmes réductibles. Elles permettent aussi de corriger les systèmes de contraintes (détecter les contraintes conflictuelles et les entités géométriques sur- et sous-contraintes).

Ce chapitre est organisé comme suit : en section 1, nous rappelons quelques définitions et notations de la théorie des graphes que nous utiliserons tout au long de ce chapitre. En section 2, nous présentons une introduction aux décompositions. En section 3, nous donnons quelques exemples de contraintes et leurs équations correspondantes. En section 4, nous donnons quelques détails sur la relation entre les systèmes algébriques et les graphes qui les modélisent. En section 5, nous analysons la structure mathématique des graphes bipartis. Nous présentons la décomposition de Dulmage-Mendelsohn (que nous noterons DM-décomposition) et celle de König en expliquant comment les obtenir. En section 6, nous détaillons les algorithmes qui calculent ces décompositions : ils sont linéaires en espace et leur temps est borné par le temps de recherche d'un

couplage maximum dans un graphe biparti. L'algorithme s'exécute en $O((m+n).n^{1/2})$ où m et n désignent respectivement le nombre d'arêtes et le nombre de sommets du graphe biparti. En section 7 nous donnons quelques exemples de problèmes résolus. Nous terminons ce chapitre par quelques problèmes ouverts.

1. RAPPELS

Nous rappelons dans ce qui suit quelques définitions et notations de la théorie des graphes que nous utiliserons tout au long de ce chapitre (voir [GM 79],[Berg 83]).

Graphe.

Un graphe G = (V, E) est le couple constitué :

1. par un ensemble de points $V = \{v_1, v_2, ..., v_n\}$
2. par une famille $E = (e_1, e_2, ..., e_m)$ d'éléments du produit cartésien

$$V \times V = \{(x,y) / x \in V, y \in V\} ;$$

Les points v_i sont appelés sommets du graphe et les e_i arcs du graphe. Pour tout arc (x,y), x est appelé extrémité initiale et y est appelé extrémité terminale. Le nombre de sommets du graphe G est appelé l'ordre de G. Un arc e = (x,x) dont les extrémités coïncident est appelé une boucle. Un p-graphe est un graphe dans lequel il n'existe jamais plus de p arcs de la forme (x,y) entre deux sommets quelconques x et y pris dans cet ordre.

Cardinalité.

Le cardinal d'un ensemble V sera noté $|V|$.

Graphe : concepts non orientés.

Dans l'étude de certaines propriétés des graphes, il arrive que l'orientation des arcs, c'est à dire la distinction entre extrémité initiale et extrémité terminale, ne joue aucun rôle. On s'intéresse simplement à l'existence ou la non-existence d'un (ou de plusieurs) arcs entre deux sommets (sans en préciser l'ordre). A tout arc (x,y), on associe la paire (x,y) appelé arête. Un

graphe est dit simple s'il est sans boucle et il n'y a jamais plus d'une arête entre deux sommets quelconques.

Successeur.

On dit que y est un successeur de x si il existe un arc ayant son extrémité initiale en x et son extrémité terminale en y. L'ensemble des successeurs de x se note :

$$\Gamma_G^+(x).$$

Prédécesseur.

On dit que y est un prédécesseur de x si il existe un arc de la forme (y, x). L'ensemble des prédécesseurs de x se note :

$$\Gamma_G^-(x).$$

Voisin.

L'ensemble des sommets voisins de x se note :

$$\Gamma_G(x) = \Gamma_G^+(x) \cup \Gamma_G^-(x).$$

L'ensemble des voisins d'un ensemble A tel que $A \subset V$ est donné par :

$$\Gamma_G(A) = \cup \Gamma_G(a) \qquad (a \in A)$$

Si $x \in \Gamma_G(A)$ et $x \notin A$, on dit que x est adjacent à l'ensemble A.

Source et puits.

On appelle source tout sommet de G n'ayant pas de prédécesseur. On appelle puits tout sommet de G n'ayant pas de successeur.

Graphe biparti.

Un graphe G = (V, E) est biparti si l'ensemble de ses sommets V peut être partitionné en deux sous-ensembles X et Y tels que deux sommets du même ensemble ne soient jamais adjacents. On le note G = (X, Y, E).

$$V = Y \cup X \text{ et } Y \cap X = \varnothing$$

Sous-graphe de G engendré par A ⊂ V.

C'est le graphe G_A dont les sommets sont les sommets de A, et dont les arcs sont les arcs de G ayant leurs deux extrémités dans A.

Graphe partiel de G engendré par B ⊂ E.

C'est le graphe $G_B = (V, B)$ dont les sommets sont les sommets de V, et dont les arcs sont les arcs de B. On élimine de G les arcs de E -B.

Chaîne de longueur l > 0.

C'est une séquence d'arcs $(e_1, e_2, ..., e_l)$ de G telle que chaque arc ei de la séquence $(2 \leq i \leq l - 1)$ ait une extrémité en commun avec l'arc précédent, et l'autre extrémité commune avec l'arc suivant. Le nombre d'arcs de la séquence est la longueur de la chaîne.

Chemin de longueur l > 0.

C'est une chaîne $(e_1, e_2, ..., e_l)$ d'un type particulier, où pour tout arc e_i $(i < l)$ l'extrémité terminale de e_i coïncide avec l'extrémité initiale de e_{i+1}.

Graphe connexe.

Un graphe est dit connexe si pour tout couple x et y de sommets, il existe une chaîne reliant ces deux points.

Composante connexe d'un graphe G.

la relation *R* telle que : x *R* y si il existe dans G une chaîne reliant x et y est une relation d'équivalence.

Les classes de cette relation d'équivalence constituent une partition de V en sous-graphes connexes de G, appelés les composantes connexes de G.

Composante fortement connexe d'un graphe G.

Considérons un graphe G = (V, E) connexe quelconque et soit la relation R' dans G telle que : x *R'* y si il existe un chemin allant de x à y et un chemin

allant de y à x. Cette relation est une relation d'équivalence. Ses classes constituent une partition de V, et s'appellent les composantes fortement connexes du graphe G.

Couplage.

Soit G = (V, E) un graphe simple. On appelle couplage un ensemble E_0 d'arêtes tel que deux arêtes quelconques de E_0 sont non-adjacentes. Si E_0 est un couplage, et si $E_1 \subset E_0$, alors E_1 est aussi un couplage.

On dit qu'un sommet x est saturé par un couplage E_0 si il existe une arête de E_0 attachée à x. Si un sommet x n'est attaché à aucune arête du couplage on dira que ce sommet est insaturé. Un couplage qui sature tous les sommets du graphe est appelé couplage parfait.

Chaîne alternée.

Si $K \subset E$ est un couplage d'un graphe G = (V, E) on appelle chaîne alternée (relativement à K) une chaîne de G dont les arêtes sont alternativement dans K et dans E - K. Une chaîne alternée augmentante est une chaîne alternée joignant deux sommets insaturés.

Couplage maximum.

Un couplage K est maximum si et seulement si il n'existe pas de chaîne alternée augmentante relativement à K (théorème de Berge [Berg 57]).

2. INTRODUCTION AUX DECOMPOSITIONS

La modélisation géométrique par contraintes aboutit à de grands systèmes d'équations. Les méthodes numériques de résolution (Newton-Raphson, bissection) sont au mieux en $O(n^3)$; les méthodes symboliques (Bases de Gröbner) sont exponentielles en temps et en espace (voir chapitre 1). Les méthodes générales sont donc coûteuses en temps de calcul pour de grands systèmes. Donc, toute méthode de décomposition du système d'équations en sous-systèmes simples à résoudre est intéressante.

Dans cette section, nous allons considérer le graphe biparti associé à un système d'équations. Ce graphe biparti possède un sommet par équation, un sommet par inconnue, et une arête entre une inconnue x et une équation y si, et seulement si, x apparaît dans l'équation y. Ce type de graphe a été déjà utilisé, entre autres, par Serrano dans [Serr 91].

Par convention, les sommets "équations" sont les éléments de l'ensemble Y, les sommets "inconnues" sont les éléments de l'ensemble X, et dans toutes les figures, les sommets "équations" sont représentés "au dessus" des sommets "inconnues".

Commençons par donner quelques définitions intuitives.

- Un système *sur-contraint* possède plus d'équations que d'inconnues ; ses équations peuvent être redondantes, ou incompatibles et donc ne conduire à aucune solution.

- Un système *sous-contraint* possède plus d'inconnues que d'équations ; il possède, en général, une infinité non dénombrable de solutions.

- Un système *bien contraint* possède autant d'équations que d'inconnues et ne contient aucun sous-système sur-contraint. Il possède donc un nombre fini de solutions.

Le graphe associé à un système *bien* (resp. *sur-*, *sous-*) *contraint* est dit *bien* (resp. *sur-*, sous-) *contraint*.

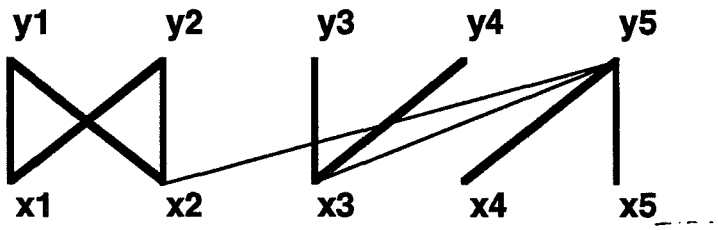


Figure 1.

Dans la figure 1, le sous-système $\{y_1, y_2, x_1, x_2\}$ est bien contraint. Le sous-système $\{y_3, y_4, x_3\}$ est sur-contraint. Le sous-système $\{y_5, x_3, x_5\}$ est sous-contraint.

Dans le cas général, ce graphe est suffisant pour décomposer le système d'équations en trois sous-systèmes: *bien contraint*, *sur-contraint*, et *sous-contraint*. Chacun de ces sous-systèmes peut être éventuellement vide. Cette décomposition existe toujours et est unique ; elle est due à Dulmage et Mendelsohn [DM 58], [DM 59], [DM 62], [DM 63].

En termes mathématiques, un système est bien contraint si, et seulement si, le graphe biparti associé satisfait à la relation de König-Hall : $|\Gamma(Y)| = |Y|$ et pour tout sous-ensemble Y' de l'ensemble Y des équations, on a $|\Gamma(Y')| \geq |Y'|$. Autrement dit, le nombre d'inconnues apparaissant dans Y' est au moins égal au nombre d'équations dans Y'. Cette condition est équivalente à l'existence d'un couplage parfait (voir plus loin).

Introduisons maintenant une deuxième décomposition.

Un graphe bien contraint G = (X, Y, E) est dit *irréductible* si, et seulement si :

- $|\Gamma(Y)| = 1$,

ou

- $|\Gamma(Y)| \neq 1$ et pour tout sous-graphe propre G' = (X', Y', E), on a : $|\Gamma(Y')| > |Y'|$.

Propriété :

Tout graphe bien contraint est soit *irréductible*, soit contient un *sous-graphe irréductible* I.

Preuve :

Un graphe G = (X, Y, E) est bien-contraint si , seulement si, $|\Gamma(Y)| = |Y|$ et pour tout sous-graphe propre G' = (X', Y', E), on a : $|\Gamma(Y')| \geq |Y'|$.

- si pour chaque sous-graphe G' on a : $|\Gamma(Y')| > |Y'|$ alors G est irréductible d'après la définition.

- si il existe un sous-graphe G' tel que $|\Gamma(Y')| = |Y'|$ alors G' est bien contraint et est soit irréductible soit contient un sous-graphe bien-

contraint. On décompose ainsi les graphes bien-contraints jusqu'à aboutir à un irréductible. Cet irréductible est aussi contenu dans G.

De plus, G - I est encore bien contraint, ou vide. Par conséquent, la résolution du système associé à G est réduite à la résolution successive des sous-systèmes associés aux sous-graphes *irréductibles* de G.

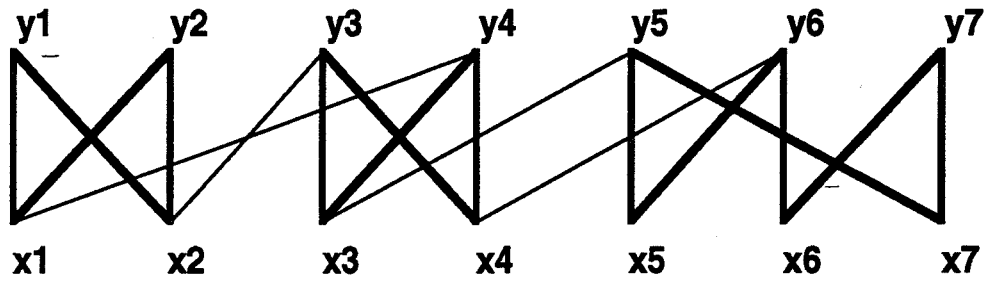


Figure 2.

Dans la figure 2, G est bien contraint. $I_1 = \{y_1, y_2, x_1, x_2\}$ est l'unique sous-graphe irréductible de G. $I_2 = \{y_3, y_4, x_3, x_4\}$ est l'unique sous-graphe irréductible de $G - I_1$. Enfin, $G - I_1 - I_2 = \{y_5, y_6, y_7, x_5, x_6, x_7\}$est irréductible.

Comme il sera prouvé par la suite, les graphes bipartis associés aux systèmes sur-contraints admettent aussi une telle décomposition, mais elle n'est pas unique.

Dans ce chapitre, nous présentons une méthode efficace pour obtenir une telle décomposition. Son coût est borné par le temps de recherche d'un couplage maximum.

Les deux décompositions décrites précédemment sont intéressantes à deux points de vue :

Premièrement, le mode de programmation par contraintes reste encore de la programmation, et par conséquent, le programmeur n'est pas affranchi de bogues! On peut même penser que le debogage de programmes de contraintes est plus difficile que celui de programmes impératifs. Dès lors, la décomposition en sous-systèmes bien, sur, ou sous-contraints constitue une première aide importante. De plus, quand le système est bien contraint, la décomposition en

sous-systèmes irréductibles permet au programmeur de suivre le processus de résolution, pas à pas : la décomposition donne la trace de l'exécution du processus de résolution. Ceci est une propriété très importante si l'on garde à l'esprit que les méthodes numériques classiques ne donnent pas aux programmeurs les détails du processus de résolution.

D'un autre côté, pour les systèmes bien-contraints, la décomposition en sous-systèmes irréductibles accélére grandement la résolution. Les méthodes numériques de résolution nécessitent des temps au moins cubiques en fonction de la taille du système, et les méthodes symboliques requièrent un temps exponentiel. Par conséquent toute méthode qui permet de réduire le système en sous-systèmes est intéressante. Même si le système est irréductible, le temps nécessaire à sa décomposition est de toute façon négligeable comparé au temps de résolution de ce système par l'approche naïve.

Pour les systèmes sur-contraints, le graphe associé n'est pas suffisant pour décider si les équations sont redondantes ou incompatibles. Une méthode possible pour résoudre un tel système est de trouver un sous-système bien-contraint quelconque, de le résoudre (ce système peut aussi être réduit en composantes irréductibles) et de vérifier si le reste des équations sont satisfaites ou non.

Pour les systèmes sous-contraints, nous n'avons pas assez d'informations pour calculer toutes les inconnues. Une méthode possible est de considérer certaines de ces inconnues comme des paramètres. On pourra les choisir de façon à obtenir un système bien-contraint, ensuite on pourra leur appliquer la méthode précédente. Cependant ce choix soulève un problème : comment obtenir la meilleure décomposition possible ; il sera traité ultérieurement.

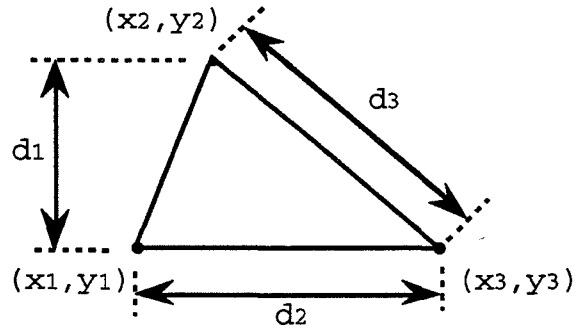
Murota [Muro 87] a utilisé la décomposition de Dulmage-Mendelsohn, mais avec une modélisation sophistiquée : il distingue inconnues et paramètres et utilise un graphe biparti différent. La décomposition de Dulmage-Mendelsohn est quelquefois utilisée pour résoudre de grands systèmes d'équations linéaires bien qu'il existe des méthodes spécifiques, plus efficaces dans le cas linéaire (voir [Duff 77], [PF 90]).

Les résultats présentés ici sont relativement classiques en théorie des graphes. Ils sont malgré tout peu répandus : ainsi Berge [Berg 83] n'en parle que

de façon allusive. Ces résultats semblent inconnus dans la communauté de la modélisation solide. Ce chapitre propose une introduction à cette théorie, et montre son intérêt pour la résolution pratique des systèmes de contraintes.

3. EQUATIONS ET CONTRAINTES

Dans la modélisation géométrique, chaque contrainte correspond à une ou plusieurs équations. La figure 3 illustre une construction plane et son système d'équations correspondant.



$eq1 : x1 = 0.$
$eq2 : y1 = 0.$
$eq3 : y3 - y1 = 0.$
$eq4 : y2 - y1 - d1 = 0.$
$eq5 : (x3 - x2)^2 + (y3 - y1)^2 - d3^2 = 0.$
$eq6 : x3 - x1 - d2 = 0.$

Figure 3.

Nous donnons dans ce qui suit quelques exemples de contraintes géométriques bi- et tri-dimensionnelles et leurs équations correspondantes.

3.1 CAS BI-DIMENSIONNEL

• L'équation correspondante à la contrainte de tangence (figure 4) entre les deux cercles C1 (de centre A et de rayon $r1$) et C2 (de centre B et de rayon $r2$) est la suivante :

$$(xB - xA)^2 + (yB - yA)^2 = (r1 \pm r2)^2.$$

• Si deux points $P1(x1,y1)$ et $P2(x2,y2)$ sont distants d'une longueur d, la contrainte entre ces deux points est exprimée par l'équation suivante :

$$(x1 - x2)^2 + (y1 - y2)^2 = d^2.$$

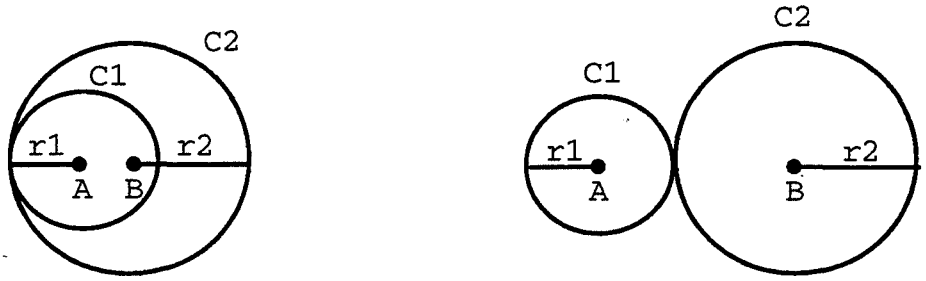


Figure 4.

• L'équation correspondante à la contrainte de distance (figure 5) entre un point $P(x_0,y_0)$ et une droite DR (d'équation : $a.x + b.y + c = 0$) est la suivante (d est la distance entre le point et la droite) :

$$(a.x_0 + b.y_0 + c)^2 - d^2.(a^2 + b^2) = 0.$$

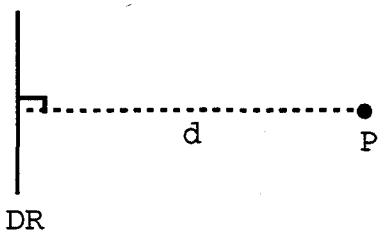


Figure 5.

• La contrainte d'angle entre deux droites DR1 (d'équation : $a_1.x + b_1.y + c_1 = 0$) et DR2 (d'équation : $a_2.x + b_2.y + c_2 = 0$) est exprimée par l'équation suivante (alpha est l'angle entre les deux droites) :

$$a_1.a_2 + b_1.b_2 = \sqrt{a_1^2 + b_1^2}.\sqrt{a_2^2 + b_2^2}.\cos(\text{alpha}).$$

3.2 CAS TRI-DIMENSIONNEL

• L'équation correspondante à la contrainte de tangence entre les deux sphères S1 (de centre A et de rayon r_1) et S2 (de centre B et de rayon r_2) est la suivante :

$$(x_B - x_A)^2 + (y_B - y_A)^2 + (z_B - z_A)^2 = (r_1 \pm r_2)^2.$$

• La contrainte d'incidence (appartenance) d'un point $P(x_0,y_0)$ à une droite DR (d'équations : $x = \lambda.a + x_d$, $y = \lambda.b + y_d$, $z = \lambda.c + z_d$) est exprimée par les équations suivantes :

$$b.(x_0 - x_d) = a.(y_0 - y_d),$$

$$c.(x_0 - x_d) = a.(z_0 - z_d).$$

• L'équation correspondante à la contrainte de distance entre un point $P(x_0,y_0)$ et un plan PL (d'équation : $A.x + B.y + C.z + D = 0$) est la suivante (d est la distance entre le point et le plan) :

$$\frac{|A.x_0 + B.y_0 + C.z_0 + D|}{\sqrt{A^2 + B^2 + C^2}} = d.$$

• La contrainte d'appartenance d'un point $P(x_0,y_0)$ à une sphère S (de centre A et de rayon r) est exprimée par l'équation suivante :

$$(x_0 - x_A)^2 + (y_0 - y_A)^2 + (z_0 - z_A)^2 = r^2.$$

4. SYSTEMES ALGEBRIQUES ET GRAPHES

Dans cette section, nous discutons les limitations de notre modélisation basée sur le fait que la connaissance des propriétés structurelles du graphe biparti associé n'est pas suffisante pour représenter toutes les propriétés du système. Mentionnons quelques-unes de ces limitations.

Un système sur-contraint a un graphe sur-contraint et inversement. Les équations d'un système sur-contraint sont soit dépendantes soit incompatibles, mais l'étude du graphe ne distingue pas ces deux cas.

Généralement un graphe sous-contraint est associé à un système qui a une infinité non dénombrable de solutions. C'est le cas du système à une équation et deux inconnues $\{x_1 + x_2 = 0\}$. Cependant certains systèmes sous contraints ont un nombre fini (ou nul) de solutions réelles, par exemple le système à une équation et deux inconnues $\{x_1^2 + x_2^2 = 0\}$. Le graphe ne donne aucun moyen de différencier ces cas.

Un graphe bien contraint est associé à un système qui posséde un nombre fini de solutions. Cependant, dans certains cas "accidentels", la matrice jacobienne peut être numériquement singulière (i.e $|\partial f_i/\partial x_j|$ est identiquement nul pour tous les x_i), et le graphe ne peut pas détecter un tel cas.

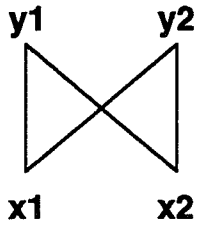


Figure 6.

Le graphe de la figure 6 peut être associé au système bien contraint $\{x_1 + x_2 = 1, x_1 + 2x_2 = 3\}$, qui a un nombre fini de solutions, au système $\{x_1 + x_2 = 1, 2x_1 + 2x_2 = 3\}$ qui n'a pas de solutions et au système $\{x_1 + x_2 = 1, 2x_1 + 2x_2 = 2\}$ qui a une infinité de solutions.

Dans les deux derniers cas de la figure 6, le jacobien est nul pour tous les x_j du fait que les coefficients satisfont à une condition "parasite" : si on perturbe légèrement certains des coefficients, sans changer le graphe biparti associé, le jacobien ne s'annule plus. Dans ces cas la nullité du jacobien est "accidentelle" et n'a rien à voir avec la structure du graphe.

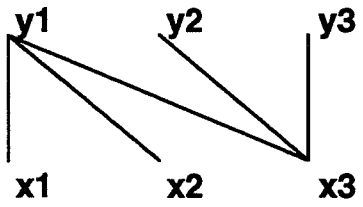


Figure 7.

En revanche, le graphe de la figure 7 n'est pas bien contraint (il n'y a pas de couplage parfait), par conséquent on peut affirmer que le jacobien de tous les systèmes associés est "structurellement" nul, et ceci quelles que soient les valeurs des coefficients. En effet, le rang du jacobien est toujours inférieur ou égal au cardinal d'un couplage maximum du graphe biparti associé. En général

ces deux valeurs sont égales. Elles sont différentes dans des cas accidentels comme mentionnés précédemment (voir [Muro 87]).

De manière plus mathématique, lorsque tous les coefficients du système sont algébriquement indépendants, alors ils ne peuvent satisfaire aucune équation parasite, et le rang du jacobien est égal au cardinal du couplage maximum (voir [Muro 87]).

5. PROPRIETES STRUCTURELLES DES GRAPHES BIPARTIS

Nous présentons maintenant les résultats fondamentaux de la décomposition DM des graphes bipartis (voir [DM 58], [DM 59],[DM 62] et [DM 63]). L'algorithme de décomposition est décrit par ce qui suit.

5.1. Décomposition de Dulmage-Mendelsohn

Comme mentionné précédemment, tout graphe biparti admet une décomposition canonique en trois parties. Le théorème suivant, dû à Dulmage-Mendelsohn, décrit cette structure.

Théorème 1 :

Soit G = (V, E) un graphe biparti quelconque. V peut être partitionné en trois ensembles D, A, et C définis par :

- *D est l'ensemble des sommets de G non saturés par au moins un couplage maximum ;*
- *A est l'ensemble des sommets de V - D, adjacents à au moins un sommet de l 'ensemble D ;*
- *C est l'ensemble V - A - D.*

Ces trois sous-ensembles sont uniques et conduisent à une décomposition unique de G en trois sous-graphes G_1, G_2, et G_3, définis par :

- $G_1 = (C_1, C_2, E_1)$, où $C_1 = C \cap Y$, $C_2 = C \cap X$, et E_1 est l'ensemble des arêtes induites par G_1 ;
- $G_2 = (D_1, A_2, E_2)$, où $D_1 = D \cap Y$, $A_2 = A \cap X$, et E_2 est l'ensemble des arêtes induites par G_2 ;

• $G_3 = (A_1, D_2, E_3)$, où $A_1 = A \cap Y$, $D_2 = D \cap X$, et E_3 est l'ensemble des arêtes induites par G_3.

Preuve :

La preuve de ce théorème, qui est assez longue, peut être trouvé dans [LP 86].

Un exemple d'une telle décomposition est montré figure 8. Notons que G_1, G_2, ou G_3 peuvent être vides. D'autre part, la DM-décomposition a été étendue aux graphes quelconques par Gallai et Edmonds [LP 86].

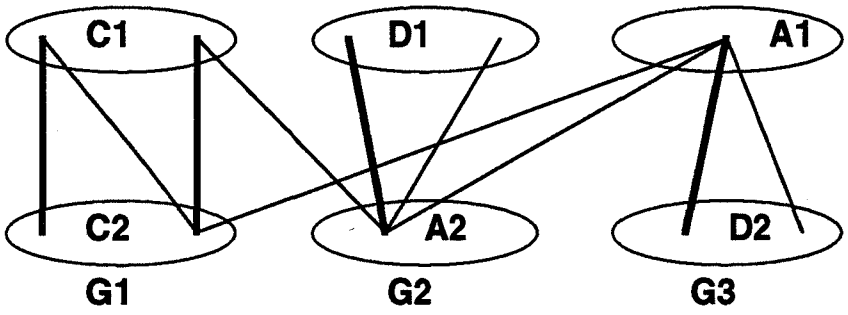


Figure 8.

Cette décomposition a les propriétés suivantes :

• Il n'existe aucune arête entre D_1 et C_2, entre D_2 et C_1, ou entre D_1 et D_2.

• G1 a un couplage parfait, donc $|C_1| = |C_2|$.

• Tout couplage maximum M de G consiste en un couplage parfait de G_1, un couplage de A_1 dans D_2 (tous les sommets de A_1 sont saturés, et au moins un sommet de D_2 ne l'est pas), et un couplage de A_2 dans D_1 (tous les sommets de A_2 sont saturés, et au moins un sommet de D_1 ne l'est pas). Donc, $|M| = |C_1| + |A_1| + |A_2|$, $|D_1| > |A_2|$, et $|A_1| < |D_2|$.

• Les arêtes entre C1 et A2, entre C2 et A1, ou entre A1 et A2, n'appartiennent jamais à un couplage maximum.

Remarque :

Dans notre modèle, G1 correspond à la composante bien contrainte du système, G2 à la composante sur-contrainte, et G3 à la composante sous-contrainte.

5.2. Graphes bipartis à couplage parfait

Soit un graphe G = (V, E) ayant un couplage parfait : le système associé est structurellement bien contraint. G est irréductible si, et seulement si, pour tout sous-graphe propre Z, on a : $|\Gamma(Z)| > |Z|$. De manière équivalente, G est irréductible si, et seulement si, pour toute arête de G, il existe toujours un couplage parfait contenant cette arête (voir [Berg 83]). En d'autres termes, avec notre modèle, un graphe irréductible correspond à un système bien contraint dont tous les sous-systèmes propres sont sous-contraints.

Un autre résultat, dû à König, et à Dulmage-Mendelsohn ([DM 58] et [DM 59]), donne l'unique décomposition canonique de tout graphe biparti ayant un couplage parfait, en sous-graphes irréductibles :

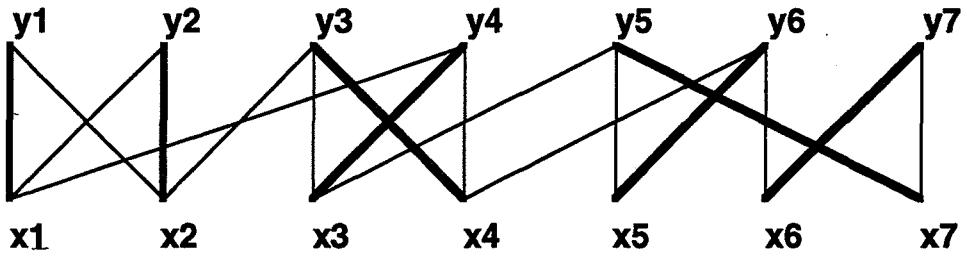
Théorème 2 :

Soit H le graphe résultant de G après avoir supprimé de G toutes les arêtes qui n'appartiennent pas à un couplage parfait de G. Alors, les composantes connexes $H_1, H_2, ..., H_k$ de H sont irréductibles.

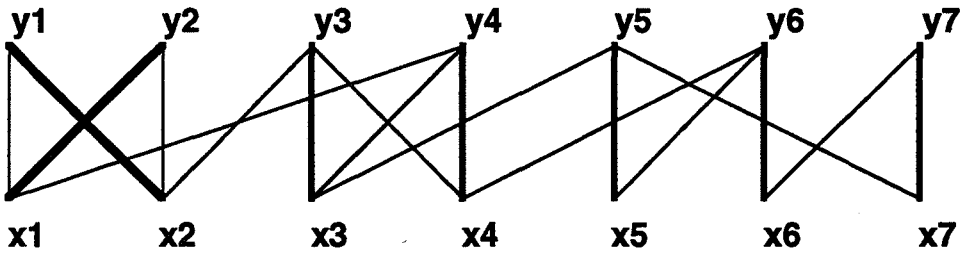
Preuve :

Soit H_i une composante connexe. Supposons que H_i ne soit pas irréductible. H_i contient alors un irréductible I (voir preuve en section deux) et H_i - I est bien-contraint. Il existe une arête A entre I et H_i - I. Cette arête n'appartient à aucun couplage parfait car autrement I ne serait pas irréductible. D'ou une contradiction avec la condition sur H_i dont toutes les arêtes appartiennent à un couplage parfait. H_i est donc irréductible.

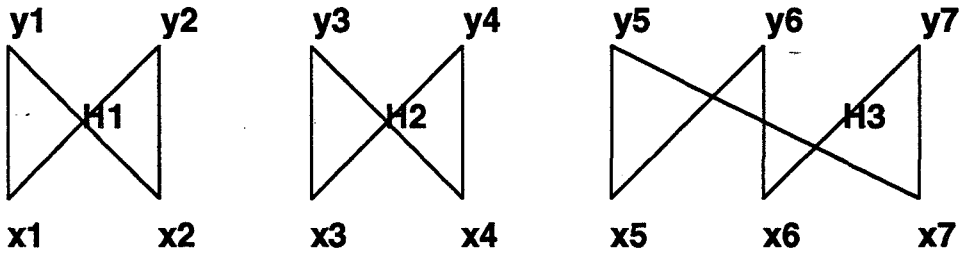
La figure 9 donne un exemple de décomposition.



Un couplage parfait d'un graphe biparti G.



Un autre couplage parfait de G.



Le graphe H.

Figure 9.

Maintenant, la question est de trouver une méthode efficace pour déterminer cette décomposition. Les propriétés suivantes donnent une réponse à cette question.

Propriété 1.

Considérons un couplage parfait M de G. Soit G' le graphe orienté obtenu à partir de G en remplaçant chaque arête xy de M par deux arcs

xy et yx, et en orientant toutes les autres arêtes de Y à X. Alors les composantes fortement connexes de G' sont exactement les composantes connexes de H : $H_1, H_2, \ldots H_k$.

Lemme :

G est irréductible ⇔ G' est fortement connexe.

Preuve :

—• Supposons G' non fortement connexe ; G' contient au moins deux composantes fortement connexes. Chaque composante correspond à un sous-graphe g = (x, y, e) de G où $|\Gamma(y)| = |y|$. G n'est donc pas un graphe irréductible.

• Supposons maintenant que G ne soit pas irréductible ; G contient alors un irréductible qui correspond à une composante fortement connexe de G' d'après la démonstration précédente. G' contient donc au moins deux composantes fortement connexes. G' n'est donc pas fortement connexe.

Preuve de la propriété 1 :

Il est suffisant de noter que nous avons : G est irréductible ⇔ G' est fortement connexe. Donc les composantes fortement connexes de G' sont irréductibles, elles correspondent nécessairement aux composantes H_1, H_2, ... H_k du théorème 2.

En fait cette construction apporte plus d'informations. Soit R le graphe orienté obtenu à partir de G en contractant chaque composante fortement connexe en un sommet de G' (ou H). Par exemple pour le graphe précédent, nous avons le graphe R donné par la figure 10.

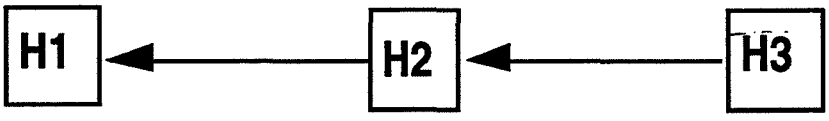


Figure 10.

Le graphe R est acyclique, donc R induit un ordre partiel sur H_1, H_2, ... H_k. Pour notre propos, si R a un arc de H_i vers H_j, alors le sous-système H_i utilise une (plusieurs) variable(s) du sous-système H_j; donc H_j doit être résolu avant H_i.

5.3. Cas général

Considérons maintenant un graphe biparti G et soit M un couplage maximum de G. Nous définissons G' comme le graphe orienté obtenu à partir de G en remplaçant chaque arête xy de M par deux arcs xy et yx, et en orientant toutes les autres arêtes de Y vers X. Les composantes fortement connexes de G' sont incluses dans G_1, dans G_2 ou dans G_3. De plus, si Y contient des sommets non saturés, alors ces sommets sont les sources de G_2 ; G_2 est donc non vide. Symétriquement, si X contient des sommets non saturés, alors ces sommets sont des puits de G_3 qui est donc non vide. Ces propriétés résultent directement de la définition de G', du théorème 1 et de ses conséquences.

Donc la structure de G' a nécessairement la forme donnée par la figure 11.

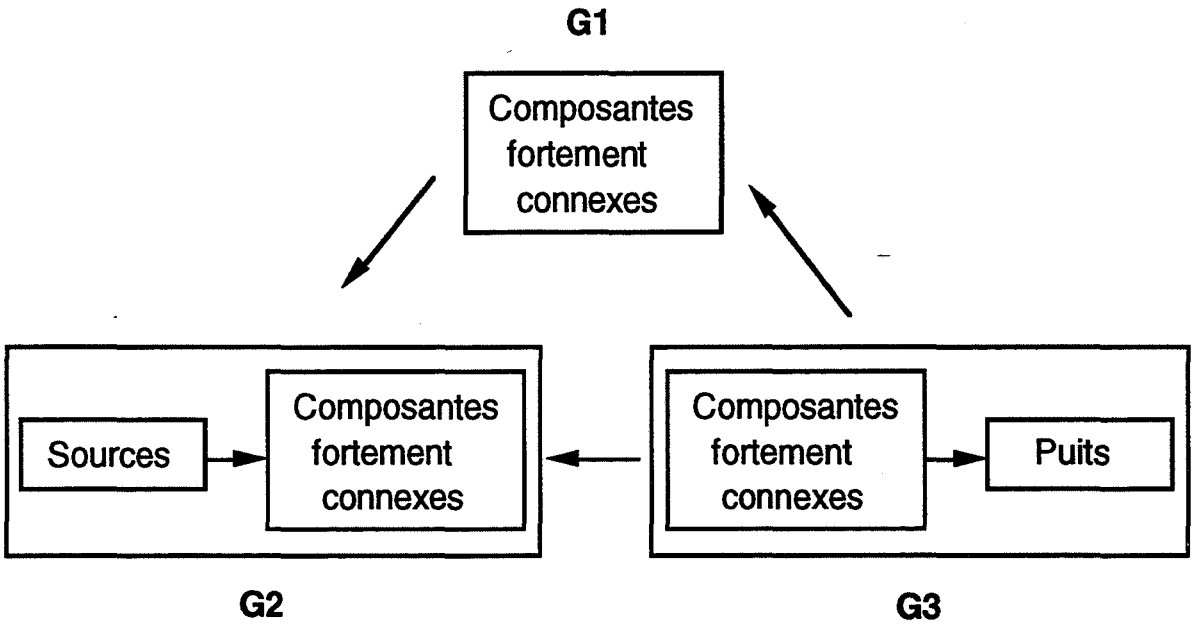


Figure 11.

La propriété 1 assure une décomposition unique de G1, mais les décompositions de G2 et G3 dépendent du couplage maximum choisi. Néanmoins on a les propriétés suivantes.

Propriété 2 :

$G_2 = \{ z \mid \exists$ chemin y,..., z dans G' tel que y est une source de G' $\}$
$G_3 = \{ t \mid \exists$ chemin t,..., x dans G' tel que x est un puits de G' $\}$

Preuve :

Conséquence immédiate de la structure de G'.

D'une façon similaire au cas des graphes ayant un couplage parfait, le graphe orienté R, obtenu à partir de G' en contractant chaque composante fortement connexe en un sommet, est acyclique. Ceci induit un ordre partiel entre les composantes fortement connexes. Donc tout ordre total compatible donne un ordre de résolution des sous-systèmes associés aux composantes fortement connexes de G1 et G2. D'autre part, G3 ne peut contenir de sous-graphe irréductible, on ne peut le résoudre que si nous affectons des valeurs à certaines inconnues correspondant à des sommets non saturés.

Il est important de mentionner que l'ordre de résolution de G_2 est fondamentalement dépendant du couplage maximum choisi. En fait, G_2 contient plusieurs sous-graphes irréductibles, et la meilleure méthode est de résoudre le plus petit en cardinalité. Cependant, la recherche efficace du plus petit sous-graphe irréductible de G_2 est un problème ouvert. Une autre stratégie est de choisir un sous ensemble S de D_1 de cardinalité $|A_2|$; le sous-graphe généré par S et noté T posséde un couplage parfait sous l'hypothèse que G_2 ait une seule composante connexe. Autrement on doit considérer chaque composante connexe de G_2. Alors nous appliquerons la décomposition donnée en 5.2 à ce sous-graphe T.

Un problème similaire survient avec G_3. Nous pouvons affecter des valeurs à tout ensemble d'inconnues de cardinalité $|D_2| - |A_1|$, par exemple les sommets non saturés de D_2. Le sous-graphe résultant T posséde un couplage parfait sous l'hypothèse que G_3 ait une seule composante connexe. Autrement on doit considérer chaque composante connexe de G_3. Alors nous appliquerons la décomposition donnée en 5.2 à ce sous-graphe T. Le problème, dans ce cas, est de trouver un bon sous ensemble S qui génére un sous-graphe T qui soit le plus petit irréductible. Ceci semble être un problème difficile.

Appliquons la décomposition mentionnée ci-dessus au graphe G de la figure 12 avec le couplage maximum $\{y_1x_1, y_2x_2, y_4x_3, y_6x_4, y_7x_6\}$:

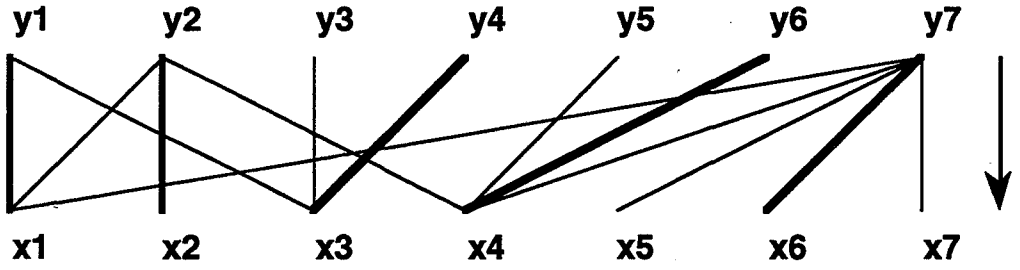


Figure 12.

y_3 et y_5 sont les sommets non saturés de Y, et donc sont les sources de G_2. D'après la propriété 2, nous avons $G_2 = \{y_3, x_3, y_4, y_5, x_4, y_6\}$, illustré par la figure 13. Notons que G_2 a deux composantes connexes $\{y_3, x_3, y_4\}$ et $\{y_5, x_4, y_6\}$.

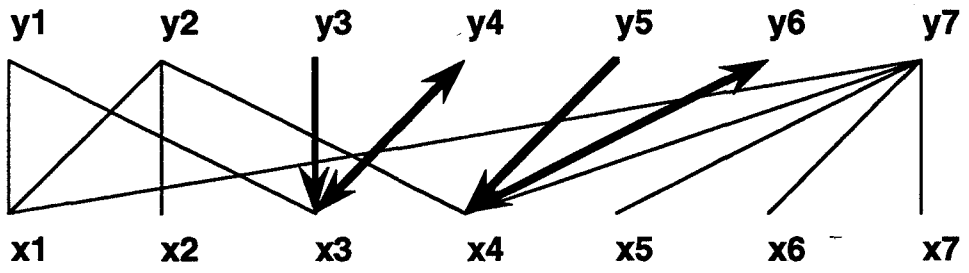


Figure 13.

x_5 et x_7 sont les sommets non saturés de X, ce sont donc les puits de G_3. D'après la propriété 2, nous avons $G_3 = \{y_7, x_5, x_6, x_7\}$, illustré par la figure 14.

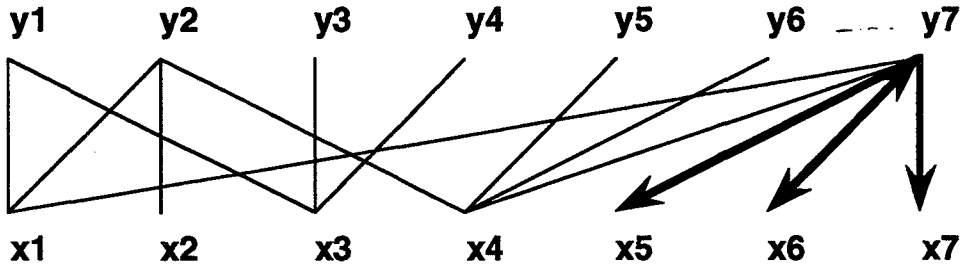


Figure 14.

Finalement, nous avons $G_1 = G - G_2 - G_3 = \{x_1, x_2, y_1, y_2\}$.

6. ALGORITHMES

Soit G = (V, E) un graphe biparti associé à un système d'équations. Notons n = |V| et m = |E|. Nous donnons dans la section suivante les algorithmes pour obtenir la décomposition présentée précédemment. Leurs preuves sont des conséquences directes des propriétés décrites dans la section 4 :

6.1. DM-Décomposition

Les sous-graphes G1, G2 et G3 de G sont obtenus à l'aide de l'algorithme suivant :

Algorithme :

1. Trouver un couplage maximum M de G.

2. Construire le graphe orienté G' à partir de G en remplaçant chaque arête xy de M par deux arcs xy et yx, et en orientant toutes les autres arêtes de Y vers X.

3. G2 est l'ensemble des descendants des sources de G'.

4. Symétriquement, G3 est l'ensemble des ancêtres des puits de G'.

5. Finalement, G1 est G - G2 - G3.

Les étapes 2, 3, 4 et 5 sont calculées en O(n + m), et l'algorithme entier s'exécute en $O((m + n).n^{1/2})$ en utilisant l'algorithme de Hopcroft et Karp [HK 73] pour l'étape 1. Cet algorithme, basé sur le théorème de Berge [Berg 57], part d'un couplage vide dont la cardinalité est augmenté d'une unité, au fur et à mesure, par la recherche de chaînes augmentantes. Ce couplage est maximal si aucune chaîne augmentante ne peut être trouvée.

Le calcul des composantes connexes de G_1, G_2 et G_3 peut être fait par une recherche en profondeur ou en largeur en un temps linéaire.

6.2. Décomposition en parties irréductibles

6.2.1 Systèmes bien contraints

Supposons que G soit bien contraint, G = G_1 et G_2 = G_3 = ∅. L'algorithme suivant donne l'unique décomposition de G en composantes irréductibles et donne un ordre de résolution entre ces différentes composantes (voir théorème 2).

Algorithme :

1. Trouver un couplage maximum M de G (dans ce cas M est un couplage parfait).

2. Construire le graphe orienté G' de G en remplaçant chaque arête xy de M par deux arcs xy et yx, et en orientant toutes les autres arêtes de Y vers X.

3. Calculer les composantes fortement connexes de G'. Chaque composante fortement connexe correspond à un irréductible.

4. Pour trouver les dépendances entre les sous-graphes irréductibles, construire le graphe R à partir de G' en contractant chaque composante fortement connexe en un sommet. Chaque arc de R allant d'un sommet s_i vers un sommet s_j a la signification suivante : résoudre le sous-système s_i avant le sous-système s_j. Un ordre total compatible entre ces sous-systèmes peut être obtenu par un tri topologique sur les sommets de R.

Les étapes 2, 3 et 4 peuvent être effectuées en O(n + m). Les étapes 3 et 4 peuvent être effectuées en utilisant l'algorithme de Tarjan [Tarj 72]. Il est à noter que l'ordre d'obtention des composantes fortement connexes (i.e les sous-systèmes irréductibles) est un ordre compatible de résolution, cet ordre est donné par l'algorithme de Tarjan. Donc l'étape 4 peut être supprimée si nous ne voulons pas d'ordre partiel induit par R. Le temps d'exécution de tout

l'algorithme en incluant l'étape 1 est borné par le coût de recherche du couplage maximum (parfait dans notre cas). Si ce couplage est déjà connu l'algorithme est linéaire.

La décomposition est indépendante du couplage maximum M.

6.2.2 Systèmes bien contraints et systèmes sur-contraints

La méthode précédente peut être appliquée à G1 ∪ G2. Cependant, le couplage maximum M n'est pas parfait, et nous avons vu que la décomposition de G2 dépend du couplage maximum M.

La méthode consiste à écarter les sommets non saturés de G2. Nous obtiendrons à ce moment là un système bien contraint qui peut être complètement résolu. A la fin, nous avons à vérifier que les solutions trouvées satisfont les équations rejetées.

7. RESULTATS

Les algorithmes décrits dans ce chapitre ont été implantés en langage LeLisp, sur une station Sun, sous le système Unix. Nous présentons dans ce qui suit quelques exemples de schémas de contraintes 2D et 3D et nous donnons pour chacun la décomposition correspondante.

7.1. Exemples 2D

Un schéma de contraintes 2D est donné par la figure 15. Les arêtes entre les sommets représentent des contraintes de dimension, ß1 et ß2 représentent les contraintes d'angles. L'ensemble des équations correspondant à ce schéma est donné par la figure 16. Les points A et B sont initialement fixés et nous notons leurs coordonnées A_x, A_y, B_x et B_y. Les coordonnées d'un point P inconnu seront notées x_P et y_P.

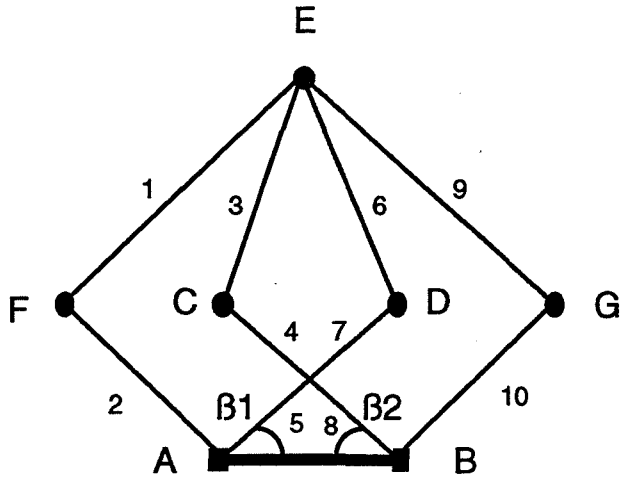


Figure 15.

eq1 : $(x_F - x_E)^2 + (y_F - y_E)^2 = d_{EF}^2$.

eq2 : $(x_F - A_x)^2 + (y_F - A_y)^2 = d_{AF}^2$.

eq3 : $(x_C - x_E)^2 + (y_C - y_E)^2 = d_{CE}^2$.

eq4 : $(x_C - B_x)^2 + (y_C - B_y)^2 = d_{BC}^2$.

eq5 : $(x_D - A_x).(B_x - A_x) + (y_D - A_y).(B_y - A_y) = d_{AD}.d_{AB}.\cos(ß1)$.

eq6 : $(x_D - x_E)^2 + (y_D - y_E)^2 = d_{DE}^2$.

eq7 : $(x_D - A_x)^2 + (y_D - A_y)^2 = d_{AD}^2$.

eq8 : $(x_C - B_x).(A_x - B_x) + (y_C - B_y).(A_y - B_y) = d_{BC}.d_{AB}.\cos(ß2)$.

eq9 : $(x_G - x_E)^2 + (y_G - y_E)^2 = d_{EG}^2$.

eq10 : $(x_G - B_x)^2 + (y_G - B_y)^2 = d_{BG}^2$.

Figure 16.

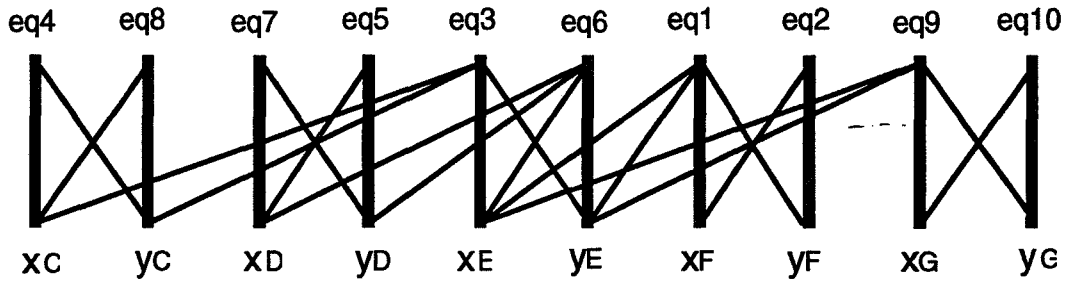


Figure 17.

Le graphe biparti G associé au système d'équations précédent et le couplage parfait {eq1xC, eq2yC, eq3xD, eq4yD, eq5xE, eq6yE, eq7xF, eq8yF, eq9xG, eq10yG} est donné par la figure 17.

Le système d'équations est bien contraint, donc G = G_1 et G_2 = G_3 = Ø. La décomposition de G en composantes irréductibles est illustrée par la figure 18. L'ordre de résolution de ces composantes irréductibles est le suivant : G_{11}, G_{12}, G_{13}, G_{14} et G_{15}. Nous calculons dans l'ordre suivant : les coordonnées du point C en utilisant les équations {eq4, eq8}, les coordonnées du point D en utilisant les équations {eq7, eq5}, les coordonnées du point E en utilisant les équations {eq3, eq6}, les coordonnées du point F en utilisant les équations {eq1, eq2} et enfin les coordonnées du point G en utilisant les équations {eq9, eq10}.

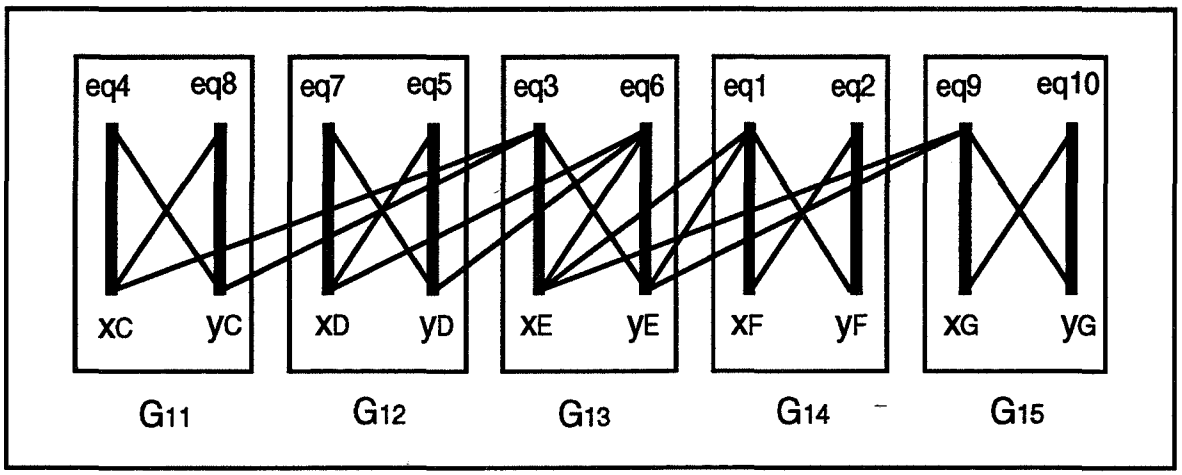


Figure 18.

Pour la résolution des systèmes algébriques, nous utiliserons la méthode de la bissection (voir chapitre 1) dite aussi méthode d'intervalles de Newton (voir [Kear 87], [MQ 82], [Snyd 92]). Cette méthode nous permet de trouver toutes les solutions d'un système algébrique dans un domaine donné. Le temps de résolution (de l'ordre de vingt-deux secondes) de la figure 15 est divisé par 20 si nous utilisons la décomposition décrite précédemment à la place de méthodes résolvant simultanément l'ensemble des équations. Le gain en temps de calcul peut être plus important pour les grands systèmes réductibles. Pour les systèmes irréductibles, la décomposition n'accélére pas la résolution mais le temps de la

décomposition (inférieur au dixième de seconde) est toutefois négligeable comparé au temps de résolution.

Les systèmes sous-contraints et les systèmes sur-contraints causent des difficultés numériques s'ils sont résolus par les méthodes classiques. La décomposition que nous utilisons ici nous permet de détecter les entités géométriques sur et sous-contraintes d'un schéma donné. Un exemple de schéma sous-contraint est donné par la figure 19. Le système d'équations correspondant est donné par la figure 20. Le point E est libre de se mouvoir le long du cercle centré en C et de rayon la distance entre les points C et E.

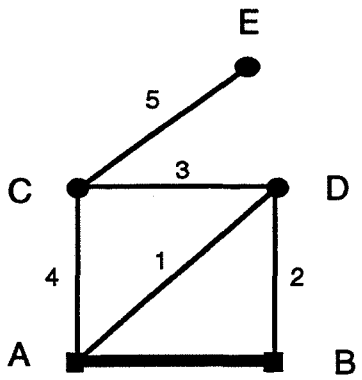


Figure 19.

$$
\begin{aligned}
eq1 &: (x_D - A_x)^2 + (y_D - A_y)^2 = d_{AD}^2. \\
eq2 &: (x_D - B_x)^2 + (y_D - B_y)^2 = d_{BD}^2. \\
eq3 &: (x_D - x_C)^2 + (y_D - y_C)^2 = d_{CD}^2. \\
eq4 &: (x_C - A_x)^2 + (y_C - A_y)^2 = d_{AC}^2. \\
eq5 &: (x_C - x_E)^2 + (y_C - y_E)^2 = d_{CE}^2.
\end{aligned}
$$

Figure 20.

La décomposition du graphe associé est donnée par la figure 21.

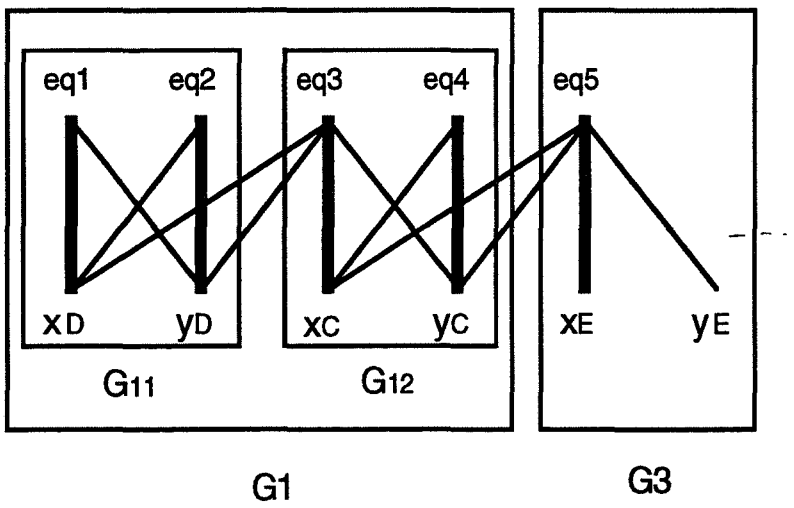


Figure 21.

$G = G1 + G3$, $G2 = \emptyset$.

Le sous graphe G3 est non vide donc nous avons une entité géométrique sous contrainte (dans notre cas le point *E*).

Un exemple de schéma sur contraint est donné par la figure 22. L'ensemble des équations correspondant à ce schéma est donné par la figure 23.

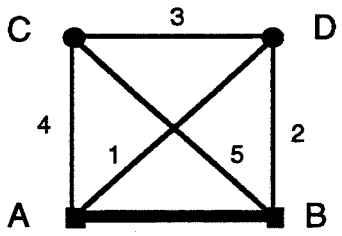
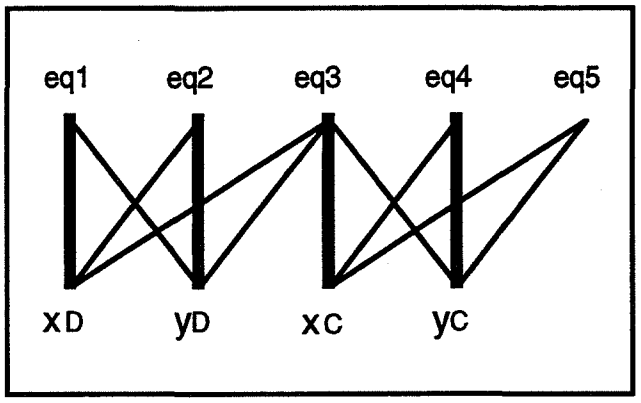


Figure 22.

eq1 : $(x_D - A_x)^2 + (y_D - A_y)^2 = d_{AD}^2$.
eq2 : $(x_D - B_x)^2 + (y_D - B_y)^2 = d_{BD}^2$.
eq3 : $(x_D - x_C)^2 + (y_D - y_C)^2 = d_{CD}^2$.
eq4 : $(x_C - A_x)^2 + (y_C - A_y)^2 = d_{AC}^2$.
eq5 : $(x_C - B_x)^2 + (y_C - B_y)^2 = d_{BC}^2$.

Figure 23.

La décomposition du graphe associé est donné par la figure 24.



G2

Figure 24.

$G = G_2$, $G_1 = G_3 = \emptyset$.

Le sous-graphe G_2 est non vide, donc nous avons des contraintes contradictoires ou redondantes. Une stratégie est de résoudre le système

d'équations après avoir enlevé l'équation (eq5) (le sommet non saturé) et de vérifier par la suite si les solutions trouvées vérifient cette équation (eq5).

7.2. Exemple 3D

La méthode de décomposition s'applique aussi bien au cas 2D qu'au cas 3D. Nous n'avons pas à traiter des cas spécifiques au cas 3D. Un schéma de contraintes 3D est donné par la figure 25 et la décomposition correspondante est illustrée par la figure 26.

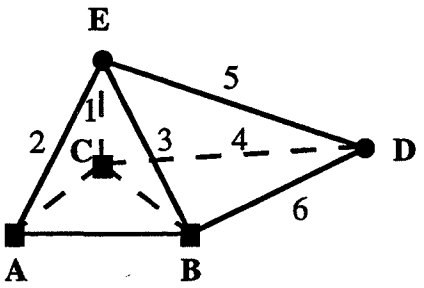


Figure 25.

Les points A, B et C sont fixés au départ. Le point E est calculé en premier (intersection de trois sphères) et le point D est calculé par la suite en utilisant les équations du deuxième sous-système.

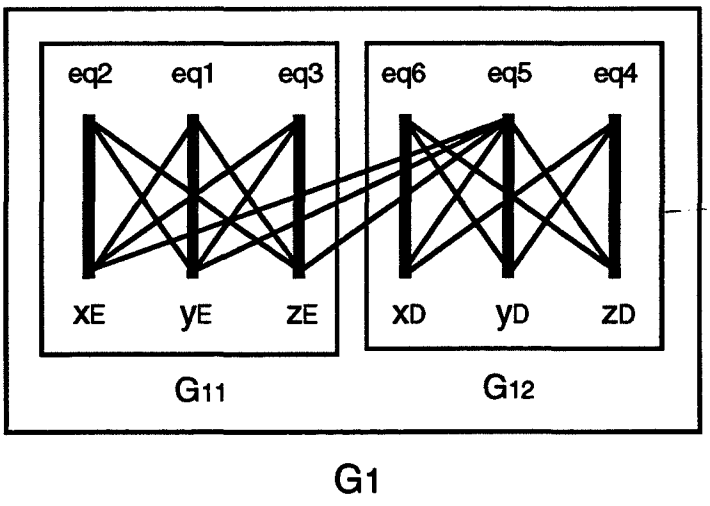


Figure 26.

8. CONCLUSION

Les méthodes présentées dans ce chapitre permettent d'accélérer le processus de résolution de systèmes d'équations et de donner les coordonnées des entités géométriques sur et sous-contraintes.

Un autre avantage de ces méthodes est qu'elles permettent de traiter un grand nombre de type de contraintes. Nous pouvons en outre facilement ajouter de nouveaux types de contraintes, il suffit pour cela d'ajouter au système à résoudre les équations relatives à ces contraintes.

Ces méthodes s'appliquent dans des espaces de dimension quelconque (en ce qui nous concerne le plan et l'espace tridimensionnel). Il n' y a pas de cas particulier à traiter propre à chaque espace. De plus l'algorithme décrit peut être utilisé pour tester la rigidité de structures 2D.

Cependant, le problème n'est pas complètement résolu pour les systèmes sur et sous-contraints. Dans le premier cas, trouver efficacement le plus petit sous-graphe irréductible de G2 à résoudre, et dans le second cas, trouver le meilleur sous-ensemble d'inconnues à affecter, semblent des problèmes intéressants de théorie des graphes.

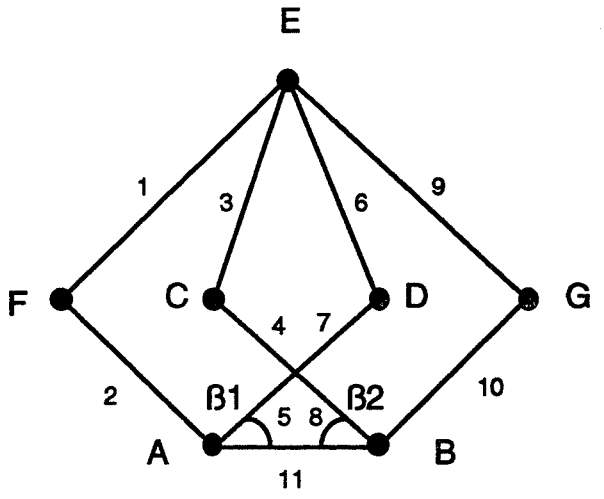


Figure 27.

Illustrons notre propos pour le cas des inconnues à affecter dans le cas des sous-contraintes. Pour cela, nous reprenons l'exemple donné par la figure 15 où cette fois-ci les coordonnées des points A et B ne sont pas connues. La contrainte de distance entre ces deux points est prise en compte.

La figure 27 est par définition sous-contrainte ; nous avons plus d'inconnues (quatorze) que d'équations (onze). Cependant cette figure est rigide à un déplacement prés car elle vérifie la condition de Laman (voir chapitre 1) i.e le nombre de contraintes est égal au double du nombre d'entités géométriques moins trois. Il faudra que nous fixions trois coordonnées pour la déterminer complètement.

L'efficacité de la méthode de décomposition, dans ce cas, dépend des coordonnées à fixer initialement. Par exemple si nous fixons les points E et F nous aurons la décomposition donnée par la figure 28.

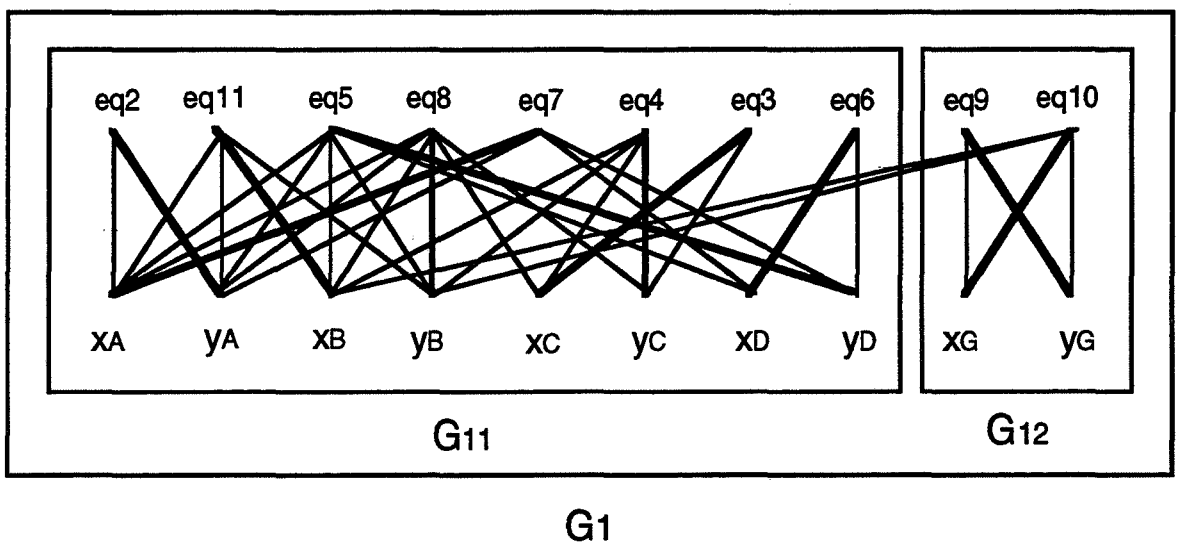


Figure 28.

Ce système d'équations se décompose en deux sous-systèmes irréductibles G_{11} et G_{12}. Le système G11 est "assez grand" (huit équations à huit inconnues à résoudre simultanément) ce qui a pour effet de "compliquer" la résolution. La meilleure décomposition possible aurait été de fixer les points A et B. Nous

aurions alors une décomposition en cinq "petits" systèmes irréductibles. La résolution de ces sous-systèmes se fait par des méthodes directes.

Dans les cas sous-contraints, trouver l'ensemble des variables à affecter pour avoir la meilleure décomposition possible reste donc un problème ouvert.

Un autre problème est que cette technique donne les irréductibles alors qu'on aurait plutôt besoin "des sous-rigides".

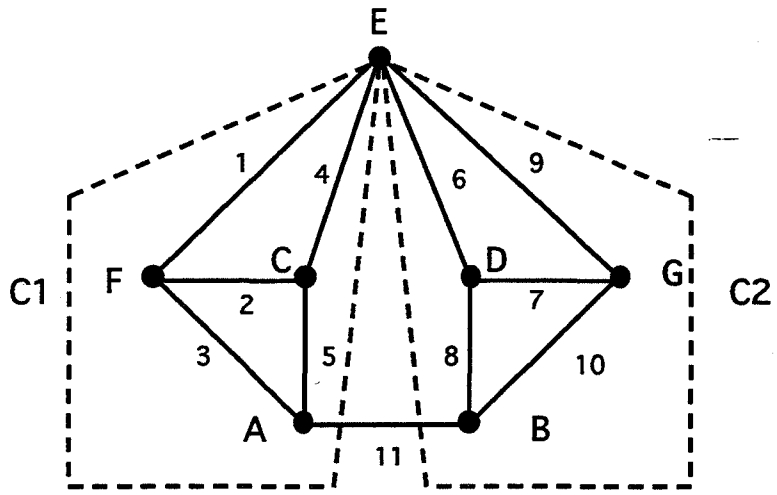


Figure 29.

Sur l'exemple de la figure 29, une, au moins, des deux composantes rigides (C1 ou C2) est résolue globalement avec la méthode de décomposition et ceci quels que soient les deux points fixés au départ (dans le cas où ce sont les points A et B qui sont fixés c'est toute la composante C1 ∪ C2 qui est résolue globalement), alors que la méthode qui semble naturelle est de résoudre la composante C1, de résoudre la composante C2 et de "recoller" ces deux composantes en tenant compte de la contrainte entre les points A et B. Nous proposerons dans le chapitre trois de cette thèse des algorithmes permettant de décomposer un problème de contraintes bi-dimensionnelles en problèmes élémentaires à résoudre et de combiner les solutions.

CHAPITRE 3

SOLVEUR GEOMETRIQUE DE CONTRAINTES

Nous présentons dans ce chapitre nos travaux sur le développement d'un système de résolution de contraintes bi-dimensionnelles par une méthode géométrique [Ait 94]. Nous étudions les différentes configurations induites par des contraintes de distances, d'angles et de tangences entre points, droites et cercles de rayon connu ou non. Ces contraintes sont représentées par un graphe. Les entités géométriques sont déterminées par un algorithme de réduction de graphes et un système à base de règles. Dans cette phase, nous procédons à la décomposition du graphe des contraintes pour trouver l'ordre de calcul des entités géométriques. Nous recherchons ensuite, pour chaque sommet du graphe, la règle applicable. Les solutions des problèmes élémentaires sont, enfin, combinées pour former la figure solution de l'ensemble des contraintes. Lorsque aucune déduction géométrique ne peut être faite, nous utilisons une méthode numérique (Bissection) pour résoudre le système de contraintes. La figure solution est donnée à un déplacement près.

Ce chapitre est organisé en six sections. La section 1 décrit notre système de façon globale. La section deux décrit les entités et les contraintes géométriques utilisées. Dans la section trois nous présentons la modélisation du problème : à partir d'un énoncé (spécifications de contraintes) de la construction géométrique introduit par l'utilisateur, un graphe des contraintes est généré. Dans la section quatre nous décrivons la méthode de résolution : cette phase utilise un algorithme de réduction de graphes et un système à base de règles. La section

cinq présente les résultats et tests. La section six décrit une méthode permettant de détecter les sous-graphes rigides d'un graphe de contraintes dans les cas bi- et tri-dimensionnels.

1. INTRODUCTION

Le solveur que nous avons développé permet de résoudre un ensemble de contraintes géométriques bi-dimensionnelles. Le principe de base de ce solveur est de décomposer le problème en problèmes élémentaires et de combiner les solutions de ces problèmes pour former la figure solution de l'ensemble des contraintes. Ce solveur accepte en entrée un énoncé (spécifications de contraintes) écrit dans un langage simple. Cet énoncé est ensuite "traduit" pour produire un "réseau" d'objets Smeci modélisant la construction géométrique. La phase résolution permettra alors la génération (en cas de succès) d'une figure géométrique vérifiant l'ensemble des contraintes.

Ce système est schématisé par la figure 1.

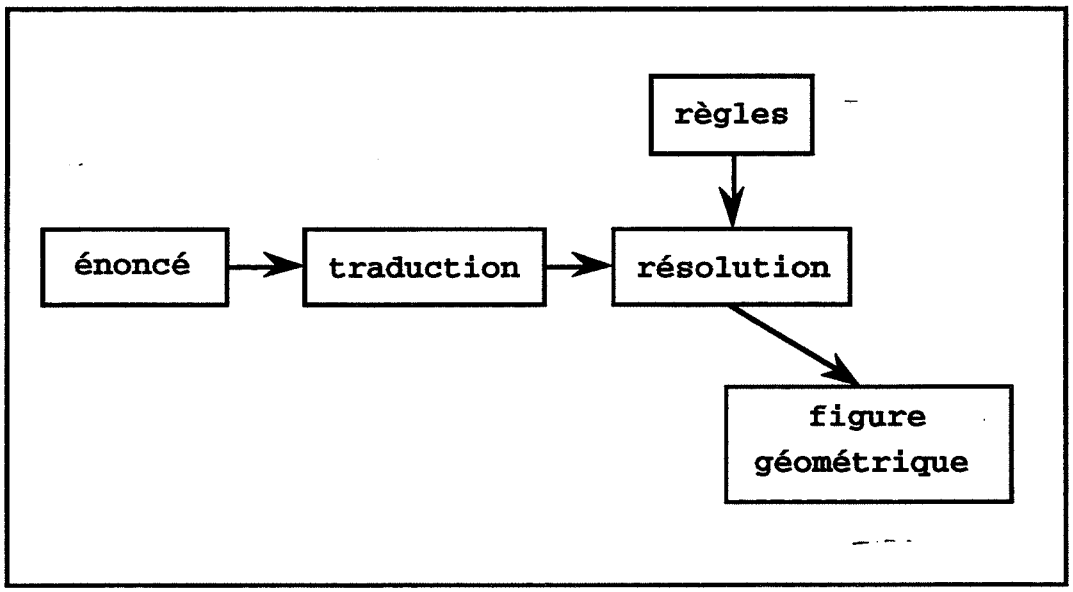


Figure 1.

La phase résolution utilise un algorithme de réduction de graphes et un système à base de règles. Les algorithmes et règles utilisés seront explicités au paragraphe 5.

2. ENTITES ET CONTRAINTES GEOMETRIQUES.

2.1. ENTITES GEOMETRIQUES ET DEGRES DE LIBERTE

Les objets géométriques élémentaires utilisés sont le point, la droite et le cercle de rayon inconnu. Le cercle de rayon connu sera traité comme un point : en effet les contraintes relatives à un cercle de rayon connu peuvent se ramener à des contraintes relatives à son centre. Par exemple, spécifier qu'un cercle de rayon connu est tangent à une droite revient à contraindre le centre de ce cercle à être distant de la droite d'une longueur égale au rayon.

En 2D, chacune des entités point et droite a deux degrés de liberté : en effet, pour fixer un point nous devons affecter des valeurs à ses deux coordonnées x et y. Une droite est complètement déterminée si nous connaissons sa pente et sa distance à l'origine. Le cercle de rayon inconnu a trois degrés de liberté.

2.2. CONTRAINTES GEOMETRIQUES

Les contraintes géométriques permettent aux utilisateurs de spécifier des propriétés géométriques sur les objets d'une scène et c'est à la charge du solveur de trouver les fonctions correspondantes applicables.

Les contraintes utilisées dans ce chapitre satisfont la condition suivante : chaque contrainte fournit des informations pour fixer un degré de liberté des objets géométriques sur lesquels elle porte. Par exemple, un point P2 non connu est contraint à être à une distance d d'un point P1 connu. Il reste encore un degré de liberté au point P2 car l'ensemble des positions possibles pour ce point est le cercle de centre P1 et de rayon d. Un objet géométrique est complètement déterminé quand il n'a plus de degré de liberté.

Définition :

> Pour éviter de futures ambiguïtés avec le degré d'un sommet du graphe de contraintes, nous parlerons de "valence" pour désigner le degré de liberté d'une entité géométrique.

L'ensemble des contraintes bi-dimensionnelles "de base" entre les entités géométriques que nous avons utilisées est le suivant :

- Contraintes de distance :
    - distance entre deux points.
    - distance entre un point et une droite.
    - distance entre un point et un cercle.

- Relation d'incidence ou contrainte de distances particulières :
    - point appartenant à une droite.
    - point appartenant à un cercle.

- Contrainte d'angle :
    - angle entre deux droites.

- Contraintes d'angles particuliers :
    - droite parallèle à une autre droite.
    - droite perpendiculaire à une autre droite.

- Contraintes d'angles particuliers :
    - cercle tangent à une droite.
    - cercle tangent à un cercle.

Les arguments d'angles et de distances sont des nombres connus. Nous avons différencié les contraintes d'angles particuliers des contraintes d'angles généraux ainsi que les contraintes de distances des relations d'incidence (ou contraintes de distance particulières) pour des raisons de commodité pour l'utilisateur et pouvoir aussi attacher aux contraintes particulières des traitements plus adaptés (plus efficaces).

3. MODELISATION DU PROBLEME : LE GRAPHE DES CONTRAINTES

3.1. ENONCE

L'utilisateur interagit avec le système en introduisant un énoncé de construction géométrique écrit dans un langage simple. Par exemple pour spécifier que deux points A et B sont distants de la longueur d, l'utilisateur écrit l'expression suivante :

(distance-point-point A B d).

Cet énoncé se décompose en deux parties :

- Déclaration des entités géométriques.
- Spécification des contraintes.

Un exemple d'énoncé est donné dans ce qui suit :

- *Déclarations.*

(Créer-Point A).
(Créer-Point B).
(Créer-Point C).
(Créer-Point D).
(Créer-Droite DRT).

- *Spécification de contraintes.*

(Distance-point-point A B 3).
(Distance-point-point A C 3).
(Distance-point-point B C 3).
(Distance-point-point D C 3).
(Point-sur-droite A DRT).
(Point-sur-droite B DRT).

3.2. TRADUCTEUR

Les entités géométriques ainsi que les contraintes bi-dimensionnelles seront représentées par des classes dans un langage orienté objet, le générateur de systèmes experts Smeci [Smec 90]. Conformément aux paradigmes de la programmation orientée objet, un objet est une instance d'une classe; la classe contient une description commune à toutes ses instances; chaque classe a ses propres méthodes, démons ...

Chaque expression de l'énoncé est traduite pour engendrer un objet Smeci de la classe correspondante. Nous donnons ci après quelques exemples de traduction :

(Créer-Point A 0 0)
(Créer-Point C)

(Distance-point-point X Y d)

Les trois expressions précédentes sont remplacées respectivement par :

(S-create-object 'point 'name A 'xy #[0 0])
(S-create-object 'point 'name C 'xy #[])
(S-create-object 'dist-pt-pt 'point1 X 'point2 Y 'distance d)

Une première phase de vérification syntaxique est faite à ce niveau. Une entité géométrique n'est créée que si le nom correspondant n'a pas déjà été affecté à une autre entité. De même une instance de contrainte géométrique n'est créée que si ses arguments existent et sont cohérents avec la contrainte spécifiée. Nous ne pouvons pas, par exemple, créer une contrainte de distance entre points avec des droites comme arguments. Il y a aussi vérification de l'argument d'angle et de distance qui doit être un nombre flottant.

Définitions :

- Un objet contraint est un objet qui contient une valeur et un ensemble (d'instances) de contraintes qui s'appliquent à cet objet.
- Une (instance de) contrainte référence les objets qu'elle contraint. Elle contient un champ "angle" ou "distance" selon la classe de la contrainte.

3.3. LIAISONS

Lors de la création d'une contrainte géométrique, il y a liaison automatique avec les entités qu'elle contraint.

Exemple :

Si deux points A et B sont liés par une contrainte de distance, la représentation de ce sous-ensemble apparaît sur la figure 2.

La correspondance entre les champs et les objets se fait par des liens inverses gérés automatiquement par Smeci. En créant la contrainte *dist-pt-pt-14* et en affectant les points A et B respectivement aux champs *point1* et *point2*, l'instance *dist-pt-pt-14* est affectée automatiquement aux champs *contr* des points A et B.

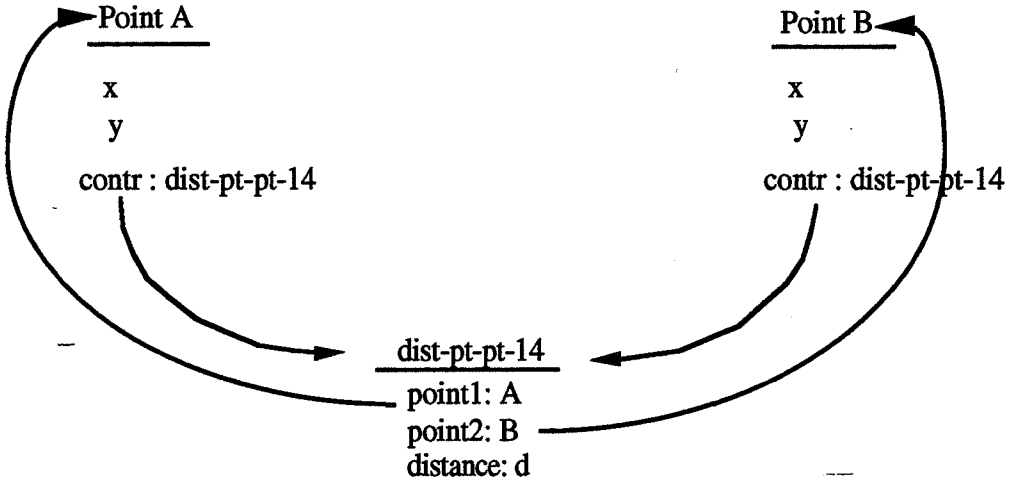


Figure 2 : Représentation de contraintes.

Après la "traduction" de l'énoncé, on obtient un graphe des contraintes. Les sommets de ce graphe représentent les entités géométriques et les arêtes représentent les contraintes géométriques.

Le graphe correspondant à l'énoncé de la section trois est donné par la figure 3.

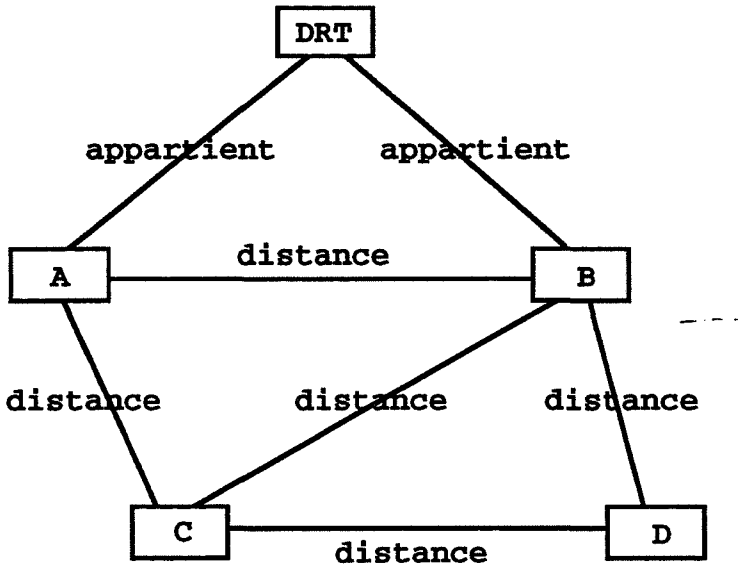


Figure 3.

4. RESOLUTION

Nous étudions dans cette section les différentes configurations induites par les contraintes de distances, d'angles et de tangences entre points, droites et cercles. Les entités géométriques sont déterminées par un algorithme de réduction de graphes et un système à base de règles. Lorsque aucune déduction géométrique ne peut être faite, nous utilisons une méthode numérique (Bissection) pour résoudre le système de contraintes. La figure solution est donnée à un déplacement près.

4.1. REGLES

Nous utilisons un ensemble de "règles" ou procédures pour résoudre des problèmes élémentaires en géométrie. Ces règles sont "déclenchées" lorsque leur partie condition est vraie. Si, par exemple, le cardinal d'un ensemble de contraintes sur une entité géométrique est égal à la valence de cette entité, on recherchera dans la base de règles, la règle applicable et on exécutera la procédure correspondante de calcul de l'entité considérée. Nous prendrons comme convention, dans cette section, de faire précéder les entités géométriques inconnues d'un point d'interrogation.

Les règles ont la structure suivante :

- **SI** <Condition> **ALORS** <Déduction>.

On distinguera trois sortes de règles. Des règles pour déterminer l'entité géométrique point, des règles pour déterminer l'entité géométrique droite et des règles pour déterminer l'entité géométrique cercle. Cette séparation nous permet une certaine optimisation lors de la recherche des règles applicables.

Nous donnons dans ce qui suit quelques exemples de règles utilisées. Les illustrations sont données pour les cas non dégénérés. Dans le cas où on a une infinité de solutions ou aucune solution pour une entité géométrique, le système signale l'inconsistance numérique à l'utilisateur. La liste complète des règles de déduction est donnée en annexe.

4.1.1. REGLES RELATIVES AUX POINTS

Nous détaillons dans ce qui suit quelques règles qui permettent de calculer les points inconnus d'une figure géométrique.

Règle R1.

<Condition>

- Un point P à déterminer est référencé par les deux contraintes suivantes :
  - point-sur-droite (?P , DRT).
  - distance-point-point (?P , A , d).

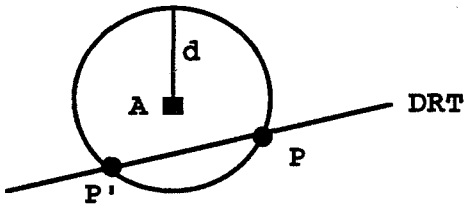


Figure 4.

<Déduction>

Le point P sera obtenu en intersectant la droite DRT et le cercle de centre A et de rayon d (voit figure 4).

Règle R2.

<Condition>

Un point P inconnu sur lequel portent les deux contraintes suivantes :
- distance-point-point (?P , A , d1).
- distance-point-point (?P , B , d2).

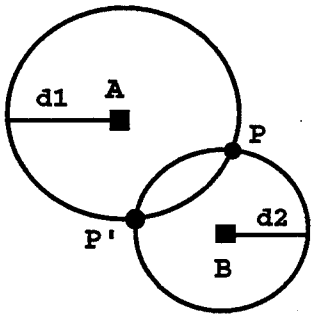


Figure 5.

<Déduction>

Le point P sera obtenu en intersectant le cercle centré en A et de rayon d1 et le cercle de centre B et de rayon d2. Nous pouvons avoir deux solutions pour le point P (voir figure 5).

Règle R3.

<Condition>

Un point P inconnu sur lequel portent les deux contraintes suivantes :
- point-sur-droite (?P , DRT1).
- point-sur-droite (?P , DRT2).

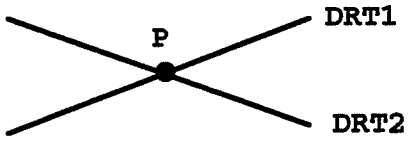


Figure 6.

<Déduction>

Le point P sera obtenu en intersectant les deux droites DRT1 et DRT2 (voir figure 6).

Règle R4.

<Condition>

Un point P inconnu sur lequel portent les deux contraintes suivantes :
- point-sur-droite (?P , DRT1).
- distance-point-droite (?P , DRT2, d).

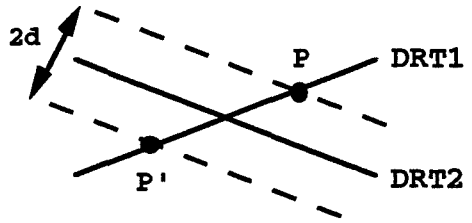


Figure 7.

<Déduction>

Le point P sera obtenu en intersectant la droite DRT1 et les deux droites parallèles à DRT2 et distantes de celle-ci de la longueur d (voir figure 7).

Règle R5.

<Condition>

Un point P inconnu sur lequel portent les deux contraintes suivantes :
- distance-point-droite (?P , DRT1, d1).
- distance-point-droite (?P , DRT2, d2).

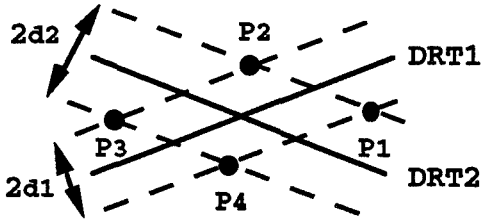


Figure 8.

<Déduction>

Le point P sera obtenu en intersectant les deux droites parallèles à DRT1 et distantes de celle-ci de la longueur d1 avec les deux droites parallèles à DRT2 et distantes de celle ci de la longueur d2 (voir figure 8).

Règle R6.

<Condition>

Un point P inconnu sur lequel portent les deux contraintes suivantes :
- distance-point-droite (?P , DRT, d1).
- point-sur-cercle (?P , CR).

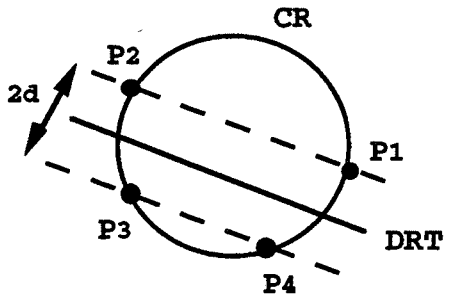


Figure 9.

<Déduction>

Le point P sera obtenu en intersectant le cercle CR avec les deux droites parallèles à DRT et distantes de celle-ci de la longueur d1 (voir figure 9).

4.1.2. REGLES RELATIVES AUX DROITES

Nous détaillons dans ce qui suit quelques règles qui permettent de calculer les droites inconnues d'une figure géométrique.

Règle R7.

<Condition>

Une droite DRT inconnue sur laquelle portent les deux contraintes suivantes :

- point-sur-droite (A , ?DRT).
- point-sur-droite (B , ?DRT).

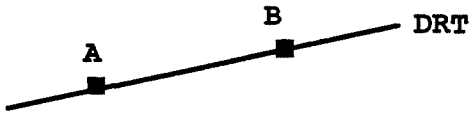


Figure 10.

<Déduction>

La droite DRT est définie par les deux points A et B (voir figure 10).

Règle R8.

<Condition>

Une droite DRT inconnue sur laquelle portent les deux contraintes suivantes :

- angle-droite-droite (?DRT, DR, a).
- point-sur-droite (A , ?DRT).

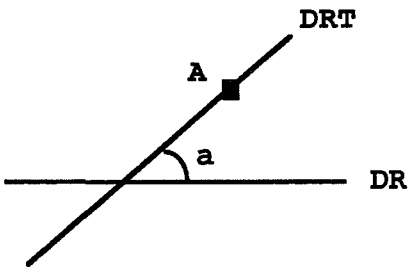


Figure 11.

<Déduction>

La droite DRT est bien définie car elle passe par le point A et fait un angle a avec la droite DR (voir figure 11).

Règle R9.

<Condition>

Une droite DRT inconnue sur laquelle portent les deux contraintes suivantes :

- point-sur-droite (A , ?DRT).
- droite-parallèle-droite (DR , ?DRT).

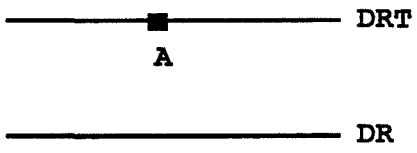


Figure 12.

<Déduction>

La droite DRT est bien définie car elle passe par le point A et est parallèle à la droite DR (voir figure 12).

Règle R10.

<Condition>

Une droite DRT inconnue sur laquelle portent les deux contraintes suivantes :

- point-sur-droite (A , ?DRT).
- cercle-tangent-droite (CR , ?DRT).

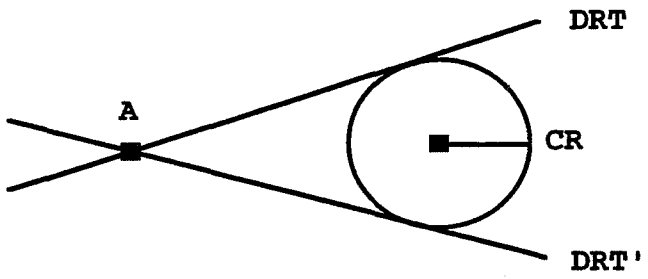


Figure 13.

<Déduction>

La droite DRT passe par le point A et est tangente au cercle CR (voir figure 13).

Règle R11.

<Condition>

Une droite DRT inconnue sur laquelle portent les deux contraintes suivantes :

- distance-point-droite (A , ?DRT, d1).
- distance-point-droite (B , ?DRT, d2).

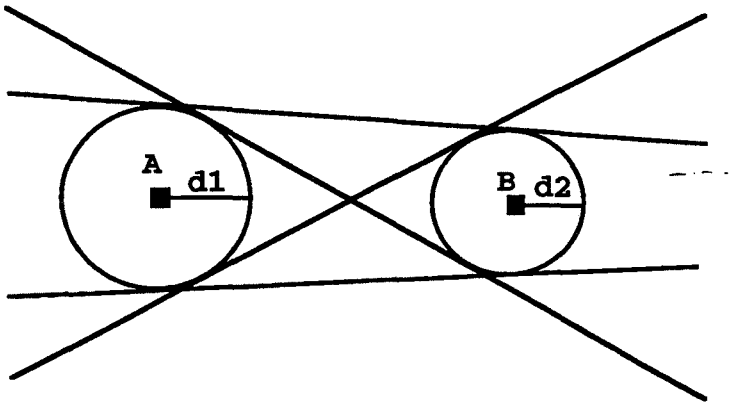


Figure 14.

<Déduction>

La droite DRT est tangente aux cercles de centres respectifs A et B et de rayons d1 et d2 (voir figure 14).

Règle R12.

<Condition>

Une droite DRT inconnue sur laquelle portent les deux contraintes suivantes :

- distance-point-droite (A , ?DRT, d).
- droite-parallèle-droite (DR , ?DRT).

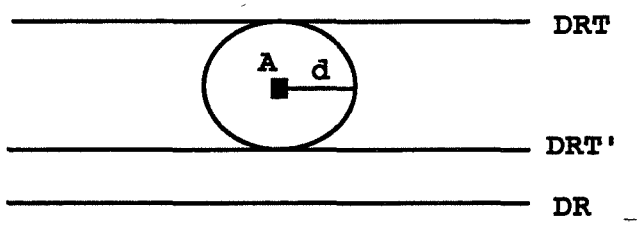


Figure 15.

<Déduction>

La droite DRT est tangente au cercle de centre A et de rayon d et est parallèle à la droite DR (voir figure 15).

Règle R13.

<Condition>

Une droite DRT inconnue sur laquelle portent les deux contraintes suivantes :

- angle-droite-droite (DR , ?DRT, a).
- distance-point-droite (A , ?DRT, d).

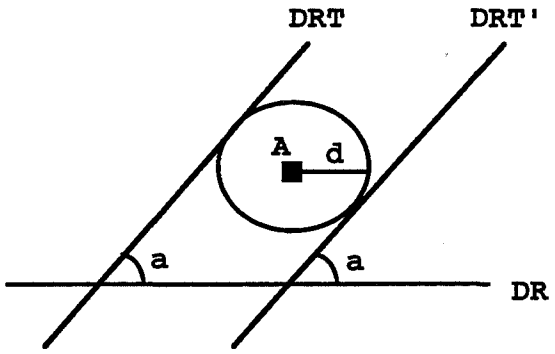


Figure 16.

<Déduction>

La droite DRT est tangente au cercle de centre A de rayon d et fait un angle a avec la droite DR (voir figure 16).

4.1.3. REGLES RELATIVES AUX CERCLES

Nous détaillons dans ce qui suit quelques règles qui permettent de calculer les cercles inconnus d'une figure géométrique.

Règle R14.

<Condition>

Un cercle CR inconnu sur lequel portent les trois contraintes suivantes :

- point-sur-cercle (A , ?CR).
- point-sur-cercle (B , ?CR).
- point-sur-cercle (C , ?CR).

<Déduction>

Le centre du cercle CR est l'intersection des médiatrices des segments AB et BC. Le rayon de ce cercle est la distance entre son centre et un quelconque des points A, B ou C. (voir figure 17).

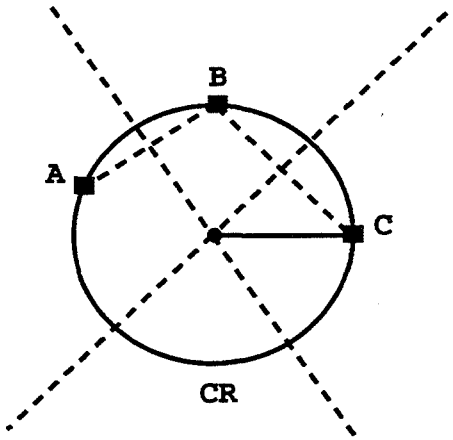


Figure 17.

Règle R15.

<Condition>

Un cercle CR2 inconnu sur lequel portent les trois contraintes suivantes :

- point-sur-cercle (A , ?CR2).
- cercle-tangent-cercle (CR , ?CR2).
- cercle-tangent-droite (?CR2, DR).

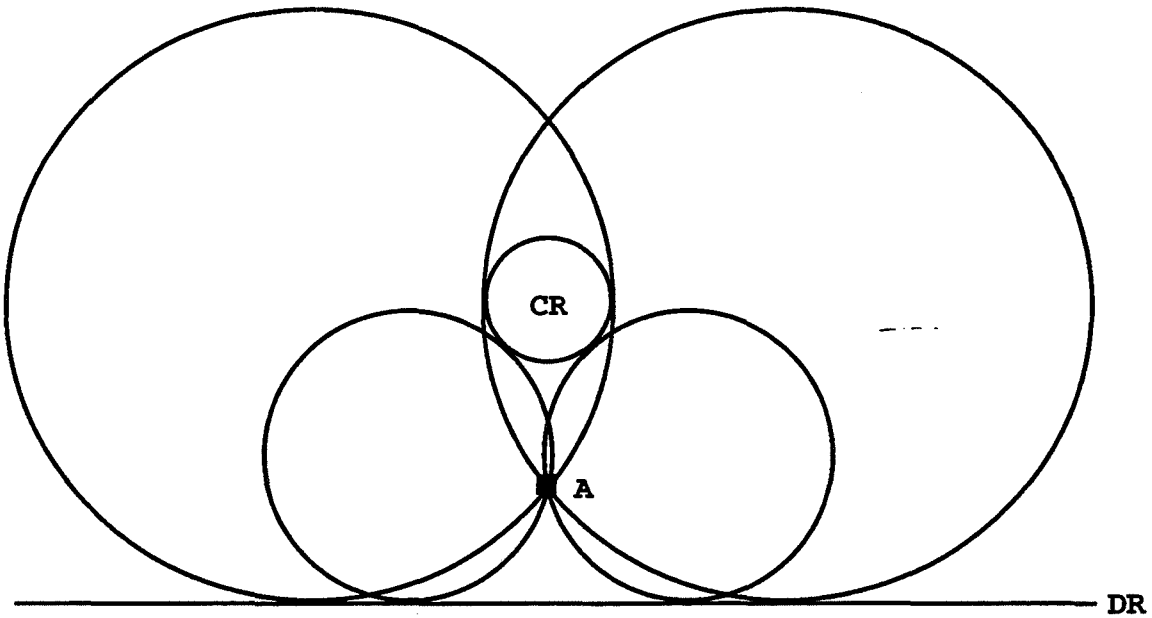


Figure 18.

<Déduction>

Nous pouvons avoir quatre solutions pour le cercle CR2 (voir figure 18).

Règle R16.

<Condition>

Un cercle CR inconnu sur lequel portent les trois contraintes suivantes :

- cercle-tangent-cercle (?CR, CR1).
- cercle-tangent-cercle (?CR, CR2).
- cercle-tangent-cercle (?CR, CR3).

<Déduction>

Nous pouvons avoir huit solutions pour le cercle CR. Pour plus de clarté, les solutions sont illustrées par les deux figures 19 et 20.

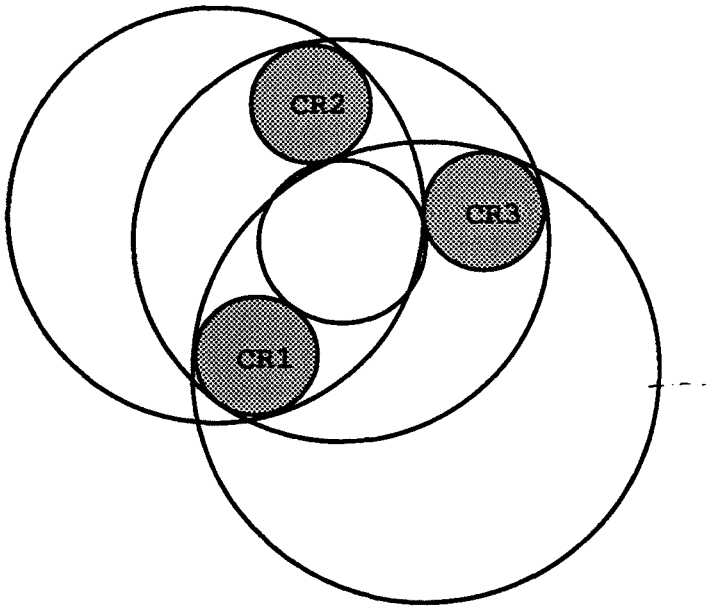


Figure 19.

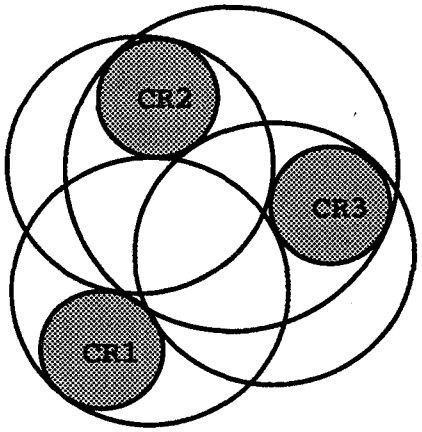


Figure 20.

Règle R17.

<Condition>

Un cercle CR inconnu sur lequel portent les trois contraintes suivantes :

- cercle-tangent-droite (?CR, DR1).
- cercle-tangent-droite (?CR, DR2).
- cercle-tangent-droite (?CR, DR3).

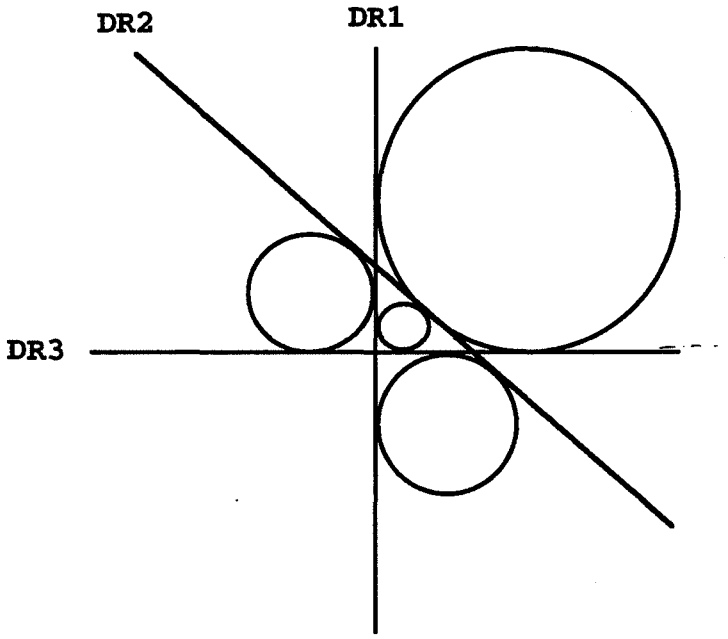


Figure 21.

<Déduction>

Nous pouvons avoir quatre solutions pour le cercle CR (voir figure 21).

Les illustrations des différentes règles ont été données pour les cas non dégénérés. Lorsqu'il existe un ensemble fini de solutions pour une entité géométrique dans les cas dégénérés, le logiciel implanté donne ces solutions. Reprenons la figure 21 où cette fois les droites DR2 et DR3 sont parallèles ; les deux solutions pour le cercle à déterminer sont données par la figure 22.

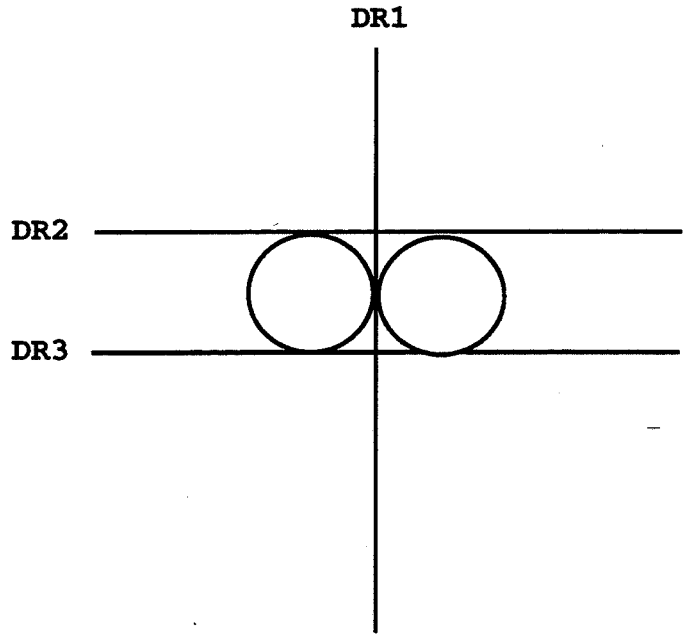


Figure 22.

4.2. ENSEMBLES D'ARTICULATION

Nous rappelons dans cette section quelques définitions des ensembles d'articulation que nous utiliserons dans le restant de ce chapitre.

Point d'articulation :

Un point d'articulation d'un graphe est un sommet dont la suppression augmente le nombre de composantes connexes de ce graphe.

Isthme :

Un isthme d'un graphe est une arête dont la suppression augmente le nombre de composantes connexes de ce graphe.

Le graphe de la figure 23 admet un point d'articulation, le sommet 4, et deux isthmes, les arêtes (1,4) et (4,5).

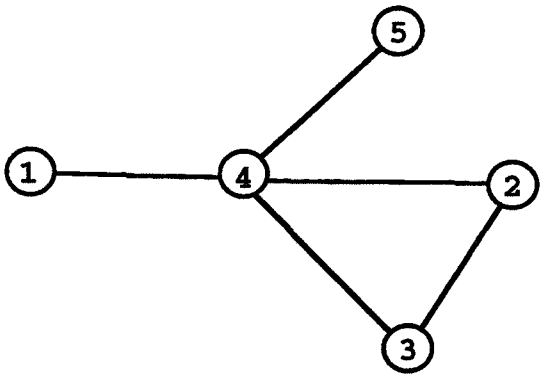


Figure 23.

2-connexité :

Un graphe G d'ordre $N \geq 3$ est 2-connexe si et seulement si :

- il est connexe,
- il n'admet pas de point d'articulation.

Les sous-graphes 2-connexes maximaux de G forment une partition des arêtes de G, et sont appelés composantes 2-connexes de G.

Ensemble d'articulation :

Un ensemble d'articulation $A \subset X$ d'un graphe connexe G est un ensemble A de sommets tel que le sous-graphe G_{X-A}, déduit de G par suppression des sommets de A, ne soit plus connexe.

k-connexité :

Un graphe G d'ordre $N \geq k+1$ est dit k-connexe si et seulement si :

- il est connexe,
- il n'admet pas d'ensemble d'articulation de cardinal k-1.

Les sous-graphes k-connexes maximaux de G forment une partition des arêtes de G, et sont appelés composantes k-connexes de G.

k-arête-connexité :

On dira qu'un graphe G connexe est k-arête-connexe s'il ne peut être déconnecté par l'élimination de moins de k arêtes.

Un graphe est donc 2-arête connexe si et seulement s'il est connexe et n'admet pas d'isthme.

4.3. DECOMPOSITION ET RESOLUTION

Dans cette phase, nous procédons à la décomposition du graphe des contraintes pour trouver l'ordre de calcul des entités géométriques. Nous recherchons ensuite, pour chaque sommet du graphe, la règle applicable. Les solutions des problèmes élémentaires sont, enfin, combinées pour former la figure solution de l'ensemble des contraintes.

4.3.1. TEST DE RIGIDITE

L'algorithme de résolution se déroule en plusieurs étapes. Avant de procéder à la résolution effective des contraintes, nous testons si le graphe est rigide. Un graphe est dit rigide (cf. chapitre 1) s'il n'admet pas de déformation continue autre que le déplacement de toute la structure. Le théorème étendu de Laman [LAMA 70] donne une condition nécessaire et suffisante pour tester la rigidité des graphes.

Théorème [LAMA 70] :

Un graphe G est rigide si et seulement si il vérifie la condition suivante :

$$C + 3 = 2.N_{sp,sd} + 3.N_{sc}$$

et si, pour tout sous-graphe G' on a :

$$C' + 3 \leq 2.N'_{sp,sd} + 3.N'_{sc}.$$

où

C est le nombre de contraintes dans le graphe,
$N_{sp,sd}$ est le nombre de sommets de type point ou droite,
N_{sc} est le nombre de sommets de type cercle,
$N = N_{sp,sd} + N_{sc}$.

La première équation de ce théorème a la signification suivante : en fixant trois coordonnées de deux sommets du graphe, le nombre d'équations (i.e contraintes) doit être égal au nombre d'inconnues (deux inconnues pour un sommet de type droite ou point et trois inconnues pour un sommet de type cercle). Les inéquations du théorème s'assurent qu'aucun sous-graphe G' n'est sur-rigide (ou sur-contraint).

Une utilisation naïve du théorème de Laman énoncé précédemment nécessite le test d'un nombre exponentiel de sous-systèmes : ce théorème ne peut être exploité pratiquement sous sa forme initiale. Plusieurs algorithmes en $O(N^2)$ furent développés pour tester la rigidité d'un graphe parmi lesquels les algorithmes de Sugihara [SUGI 80] et de Hendrickson [HEND 90]. Pour notre part, nous utiliserons l'algorithme de Hendrickson.

4.3.2. PHASE DE RESOLUTION

Cette deuxième phase s'applique aux graphes rigides. La phase de résolution proprement dite se déroule en plusieurs étapes. Ces différentes étapes exploitent la structure du graphe des contraintes pour retrouver l'ordre de résolution des entités géométriques.

4.3.2.1. GRAPHE 1-RIGIDE

Définition :

Un graphe de contraintes G rigide sera dit 1-rigide si :

- G est d'ordre deux,

ou bien :

- G est d'ordre supérieur ou égal à trois, il existe un sommet S de G tel que la valence de S est égale au degré de S et G - S est un 1-rigide.

Remarque :

Les sommets dont le degré est égal à la valence sont appelés des sommets "faciles" (car il est facile de déterminer les coordonnées et rayon correspondants par l'application d'une règle).

ETAPE 1 :

Cette étape s'applique aux graphes 1-rigides. Elle est basée sur la remarque suivante : chaque sommet qui est connecté au graphe par exactement un nombre de contraintes égal à son degré de liberté peut être calculé une fois que les sommets restants du graphe sont calculés.

Algorithme :

L1 := ∅ ;

Parcourir tous les sommets du graphe G et ajouter, au fur et mesure, en début de la liste L1 tout sommet facile ;

Tant que
il y a un sommet S dans L1
Faire
L1 := L1 - S ;
empiler (S) ;
Ajouter en fin de liste L1 tous les voisins de S qui seront des sommets faciles après la suppression de S dans G ;
Fait ;

Si il reste une arête dans le graphe G ;
Alors fixer cette arête ; /* défini plus loin */
Sinon Appliquer l'étape 2 à G ;
fsi ;

Tant que
pile non vide
Faire
dépiler (S) ;
calculer (S) ;
Fait ;

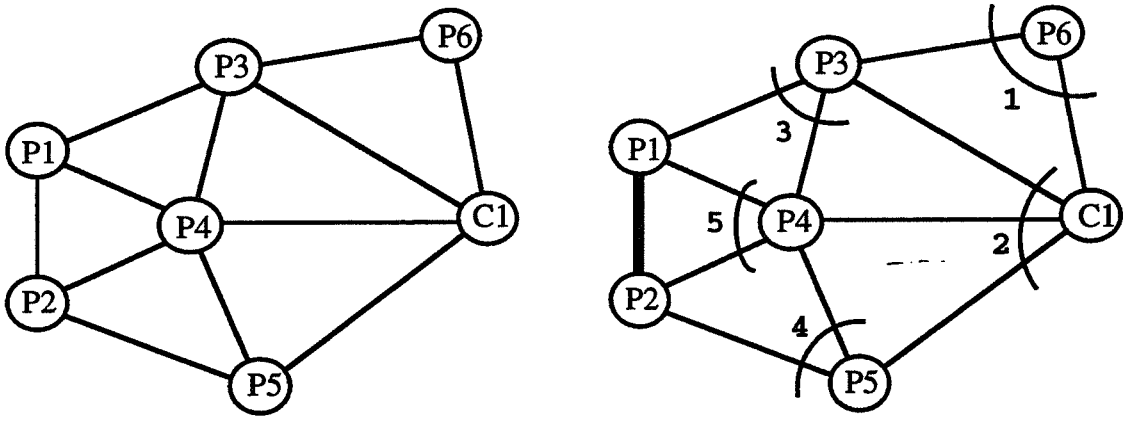
Fixer une arête (contrainte) revient à placer dans le plan les deux sommets reliés par cette arête. Nous donnons dans ce qui suit quelques exemples de placement des deux sommets en fonction de la contrainte utilisée :

- (distance-point-point A B d) : Nous affectons les coordonnées (0, 0) au point A et (d, 0) au point B.
- (point-sur-droite A DR) : Le point A est placé à l'origine est la droite DR est confondue avec l'axe de abscisses.
- (angle-droite-droite DR1 DR2 ß) : La droite DR1 est confondue avec l'axe des abscisses et la droite DR2 passe par l'origine et a pour pente tangente(ß).

Pour calculer un sommet S du graphe G, une règle de la base est déclenchée selon le type des contraintes qui référencent ce sommet. Cette étape s'exécute en un temps linéaire.

Exemple :

Soient P1, P2, P3, P4, P5, P6 des points, C1 un cercle et soit G le graphe suivant des contraintes concernant ces entités. Les arêtes entre les points représentent des contraintes de distances et les arêtes entre les points et le cercle représentent des contraintes d'appartenance (point appartenant à cercle).



Graphe G Réduction

Figure 24.

La numérotation des sommets donne l'ordre d'empilement de ces derniers. La résolution se fera dans l'ordre inverse.

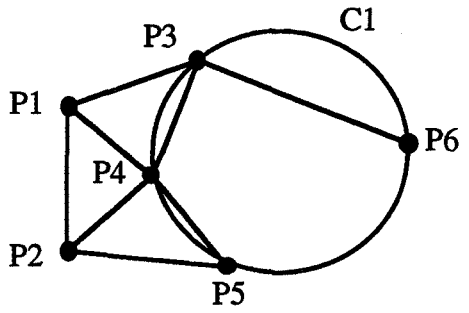


Figure 25 : une figure solution pour l'exemple précédent.

4.3.2.2. GRAPHE 2-RIGIDE

S'il n'existe pas de sommets faciles dans le graphe de contraintes, l'étape 1 ne s'applique plus. C'est pourquoi nous introduisons la notion de graphe 2-rigide.

Définition :

> Un graphe de contraintes G rigide sera dit 2-rigide s'il est composé de deux, ou plusieurs composantes 1-rigides reliées entre elles par des ensembles de trois arêtes (i.e contraintes).

Des exemples de graphes 2-rigides sont donnés par les figures 26 et 27. Les graphes Gi sont 1-rigides et leurs arêtes ne sont pas tracées.

Pour résoudre un graphe 2-rigide, nous procédons à la recherche des arêtes d'articulation, nous résolvons les composantes 1-rigide de ce graphe et nous assemblons ces composantes. Mais avant de traiter le cas général des graphes 2-rigides, nous allons considérer le cas, plus simple, des graphes 2-rigides dégénérés : dans cette classe de graphe, deux des trois arêtes d'articulation peuvent être adjacentes.

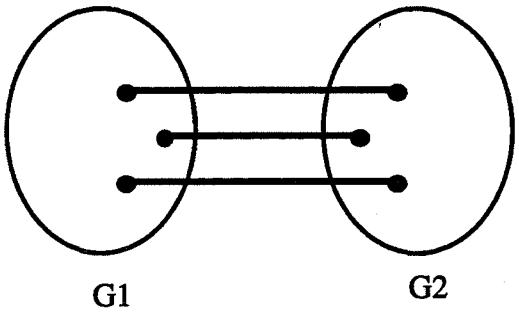


Figure 26.

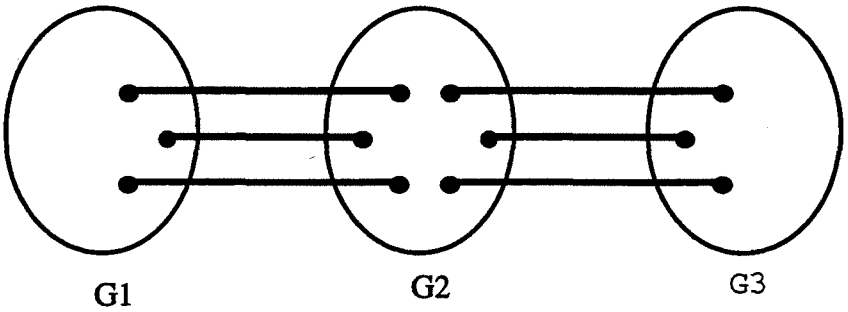


Figure 27.

Un exemples de graphes 2-rigide dégénéré est donné par la figures 28. Deux des trois arêtes d'articulation du graphe de la figure 28 sont adjacentes ; les sommets 4 et 5 de ce graphe forment un ensemble d'articulation de cardinal deux ou paire d'articulation.

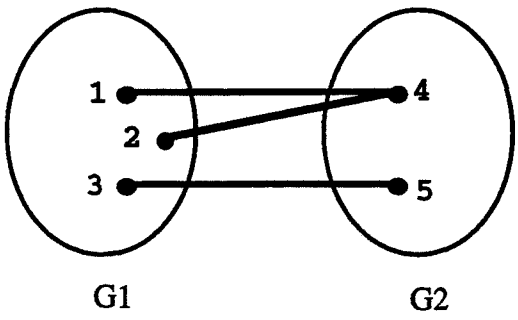


Figure 28.

Définition :

Tout graphe 2-rigide dégénéré admet (au moins) deux sommets d'articulation (ou paire d'articulation).

ETAPE 2 :

Cette étape s'applique aux graphes 2-rigides dégénérés. Ces graphes pouvant être décomposés en deux ou plusieurs sous-graphes ayant deux sommets en commun et dont l'un (au moins) est rigide. Nous chercherons donc à trouver deux sommets d'articulation de G. Cette étape est similaire, d'une certaine façon, à la méthode de Owen [Owen 91], présentée au chapitre un. Cependant Owen ne considère que des entités géométriques de type point ou droite.

L'algorithme de résolution est le suivant :

Algorithme :

Si
il existe une paire d'articulation de G
Alors
Séparer G en G1 et G2 ;
/* soient G1' et G2' les sous-graphes de G obtenus après la suppression des deux sommets d'articulation ; G1 et G2 sont les sous graphes de G dont les sommets sont respectivement les sommets de G1' et G2' plus les deux sommets d'articulation. */
Si
au moins une de ces deux composantes est rigide
Alors
Soit G1 la composante rigide ;
Appliquer étape 1 à G1 ;
Si
G2 est non-rigide
Alors
Reporter la contrainte de G1 entre les deux sommets d'articulation dans G2 ; /* G2 devient alors rigide */
fsi ;

Appliquer l'étape 1 à G2 ;
Recoller G1 et G2 ;
Sinon
Appliquer l'étape 4 à G ; /* détaillé ultérieurement */
fsi ;
Sinon
Appliquer l'étape 3 à G ;
fsi ;

La recherche d'une paire d'articulation se fait en un temps linéaire en utilisant l'algorithme de Hopcroft et Tarjan [HOPC 73]. L'étape 2 s'exécute alors en un temps linéaire.

Exemple :

Soient P1, P2, P3, P4, P5, P6, P7, P8, P9 des points et C1 un cercle et soit G le graphe des contraintes concernant ces entités.

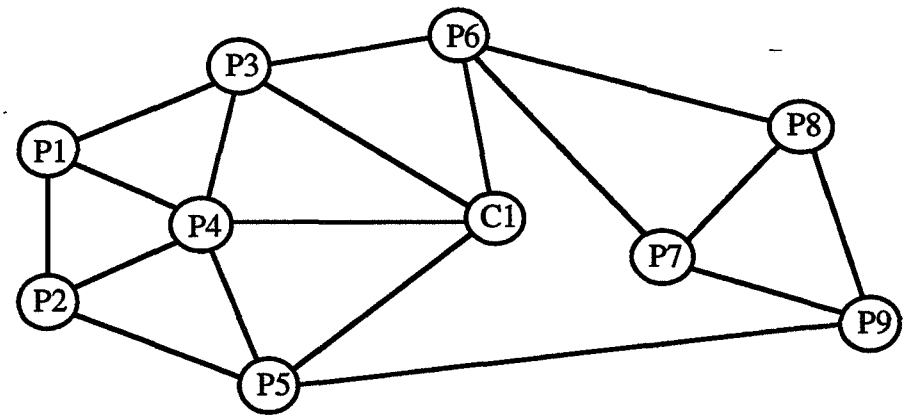
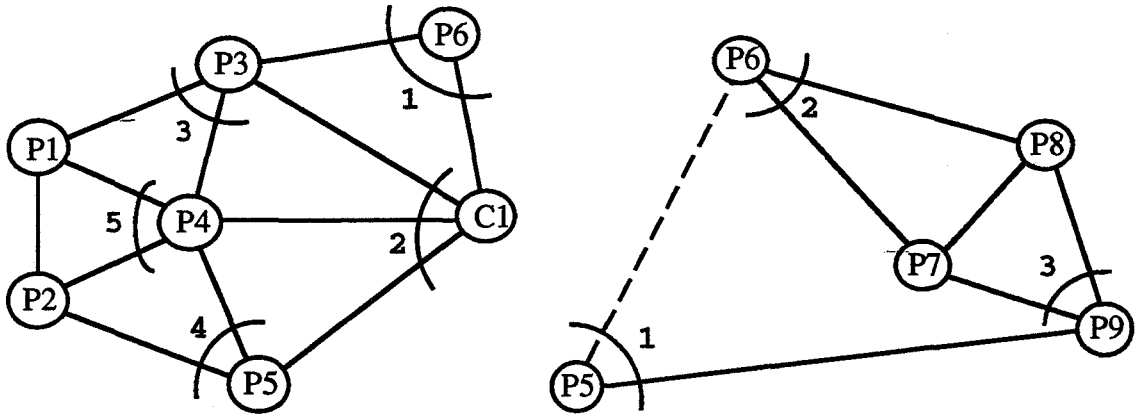


Figure 29 : Graphe G

Il n'existe aucun sommet facile S dans G. Par contre P5P6 est une paire d'articulation. Décomposons G en G1 et G2. G1 est rigide, G2 ne l'est pas. Nous résolvons G1, nous en déduisons la distance d entre P5 et P6. Nous ajoutons une contrainte : P5P6 = d au graphe G2, qui devient le graphe rigide G2' ; nous

résolvons G2'. Une fois G2' résolu, nous calculons les paramètres de la translation et de la rotation à appliquer aux sommets de G2' pour faire correspondre les deux sommets en commun des deux graphes, en l'occurrence P5 et P6.



Graphe G1 Graphe G2'

Figure 30.

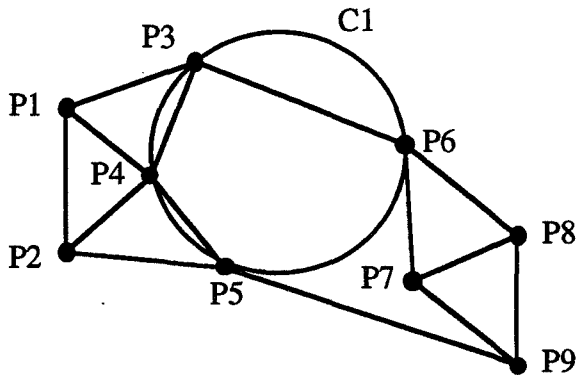


Figure 31 : Une figure solution pour l'exemple précédent.

Remarques :

• Si les deux sommets de la paire d'articulation sont des points, au moins une des deux composantes du graphe est rigide car si les deux composantes étaient non-rigides, le graphe G serait non-rigide, d'où une contradiction avec l'hypothèse sur G.

• Si au moins un des sommets de la paire d'articulation est un cercle de rayon inconnu, il peut arriver que les deux composantes G1 et G2 soient non-rigides. Un tel cas est donné par la figure 32. P5C1 est une paire d'articulation et les deux sous-graphes sont non-rigides. Dans ce cas la résolution est faite globalement par la méthode de la bissection (voir étape quatre). Par contre la figure 33 illustre le cas d'une paire d'articulation P3C1 comprenant un cercle de rayon inconnu et où un des deux sous-graphes est bien rigide (P1 P2 P3 C1).

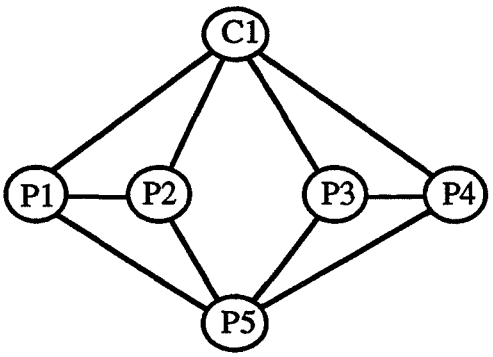


Figure 32.

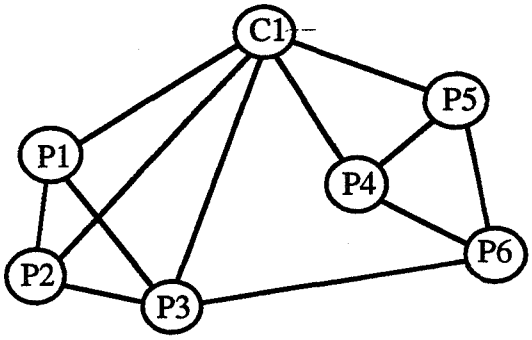


Figure 33.

ETAPE 3 :

Dans le cas où les trois arêtes d'articulations sont non adjacentes l'étape 2 ne s'applique plus. Cette étape s'applique aux graphes rigides composés de deux composantes G1 et G2 de cardinalité supérieure à un reliées par trois contraintes. Nous résolvons les deux composantes rigides ; nous calculons les paramètres du déplacement (translation et rotation) à partir des trois contraintes reliant les deux composantes ; nous recollons les deux composantes rigides en appliquant le déplacement à l'une de ces composantes.

Théorème :

G1 et G2 sont rigides.

Preuve :

Soit G un graphe rigide ayant C contraintes, $N_{sp,sd}$ sommets de type point ou droite et N_{sc} sommets de type cercle et soient G1 et G2 les deux

composantes de G reliés par trois contraintes. G1 a C1 contraintes, $N1_{sp,sd}$ sommets de type point ou droite et $N1_{sc}$ sommets de type cercle. G2 a C2 contraintes, $N2_{sp,sd}$ sommets de type point ou droite et $N2_{sc}$ sommets de type cercle.

$N_{sp,sd} = N1_{sp,sd} + N2_{sp,sd}$.
$N_{sc} = N1_{sc} + N2_{sc}$.
$C = C1 + C2 + 3$.

G est rigide donc $C + 3 = 2.N_{sp,sd} + 3.N_{sc}$ et pour tout sous-graphe G' on a : $C' + 3 \leq 2.N'_{sp,sd} + 3.N'_{sc}$. Donc $C1 + 3 \leq 2.N1_{sp,sd} + 3.N1_{sc}$ et $C2 + 3 \leq 2.N2_{sp,sd} + 3.N2_{sc}$. Supposons que G1 soit sous-contraint : $C1 + 3 < 2.N1_{sp,sd} + 3.N1_{sc}$.

En additionnant :

$C1 + 3 < 2.N1_{sp,sd} + 3.N1_{sc}$.
$C2 + 3 \leq 2.N2_{sp,sd} + 3.N2_{sc}$.

On aboutit à :

$C1 + 3 + C2 + 3 < 2.N1_{sp,sd} + 3.N1_{sc} + 2.N2_{sp,sd} + 3.N2_{sc}$.
$C + 3 < 2.(N1_{sp,sd} + N2_{sp,sd}) + 3.(N1_{sc} + N2_{sc})$.
$C + 3 < 2.N_{sp,sd} + 3.N_{sc}$.

d'où une contradiction avec l'hypothèse sur G. G1 est donc rigide. Par symétrie, on en déduit que la composante G2 est rigide.

Algorithme :

Si

il existe trois arêtes dont la suppression sépare G en deux composantes G1 et G2 (voir figure 27 ; la figure 34 donne l'exemple le plus simple) ;

Alors

/* G1 et G2 sont rigides */
Résoudre G1 ;
Résoudre G2 ;
Fixer la composante ayant le plus grand nombre d'entités géométriques;
Soit G1 cette composante ;

Calculer les paramètres du déplacement D (translation et rotation) à partir des trois arêtes d'articulations.

Replacer G2 à l'aide du déplacement D ;

Sinon

Appliquer l'étape 4 à G ;

fsi.

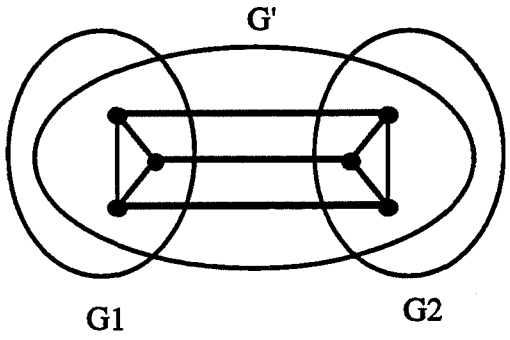


Figure 34.

Cette étape s'exécute en un temps T donné par les relations suivantes :

$$Cste.N \leq T \leq Cste.N^3.$$

Si la recherche des trois arêtes dont la suppression sépare G en deux composantes aboutit dès le premier test, cette phase est constante. Par contre si la recherche aboutit après avoir testé toutes les arêtes trois à trois, cette phase est en $O(N^3)$.

Le calcul des paramètres du déplacement D revient à résoudre, en fait, un système de trois équations à trois inconnues. En effet, après avoir résolu G1 et G2 nous transformons symboliquement les points S4, S5 et S6 (points adjacents aux trois arêtes d'articulations) du graphe G2 (voir figure 35) par le déplacement $D = T \circ R$ où T est une translation de vecteur (t_x, t_y) inconnu et R une rotation d'angle θ inconnu ; nous obtenons alors les points S1', S2' et S3'. Ces points doivent vérifier les contraintes représentées par les arêtes entre G1 et G2. De ces trois équations nous pouvons déduire les paramètres t_x et t_y de la translation T et l'angle θ de la rotation R à appliquer aux sommets de G2 pour recoller G1 et G2.

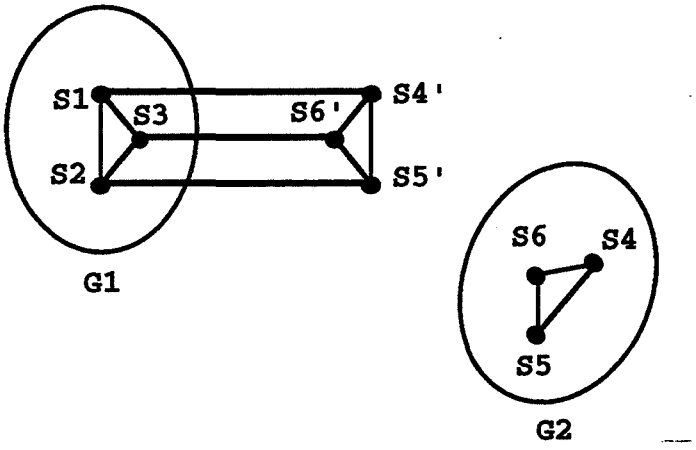


Figure 35.

Le déplacement T o R est défini par les équations suivantes :

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

S4' = T o R (S4).
S5' = T o R (S5).
S6' = T o R (S6).

En posant : $t = \frac{\tan\theta}{2}$,

nous avons :

$$\cos\theta = \frac{1 - t^2}{1 + t^2},$$

$$\sin\theta = \frac{2.t}{1 + t^2}.$$

Le système défini par les trois contraintes entre G1 et G2 est un système de trois équations à trois inconnues : t_x, t_y et t.

Exemple :

Soient P0 , P1, P2, P3, P4, P5, P6, P7, P8 et P9 des points et soit G le graphe des contraintes concernant ces entités. Les arêtes entre les points représentent des contraintes de distances .

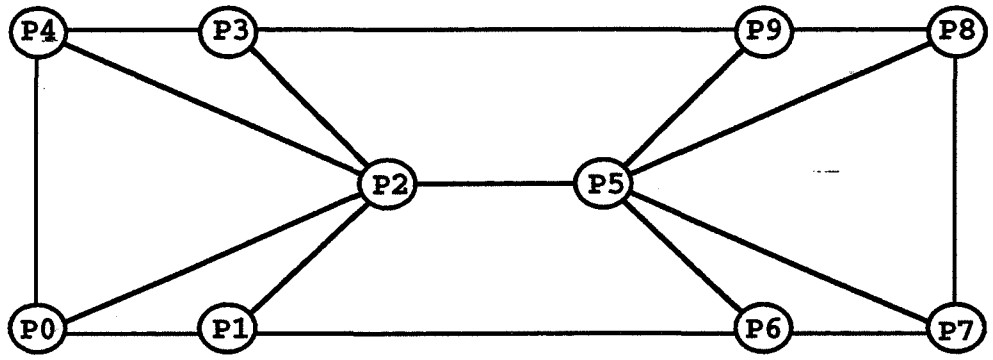
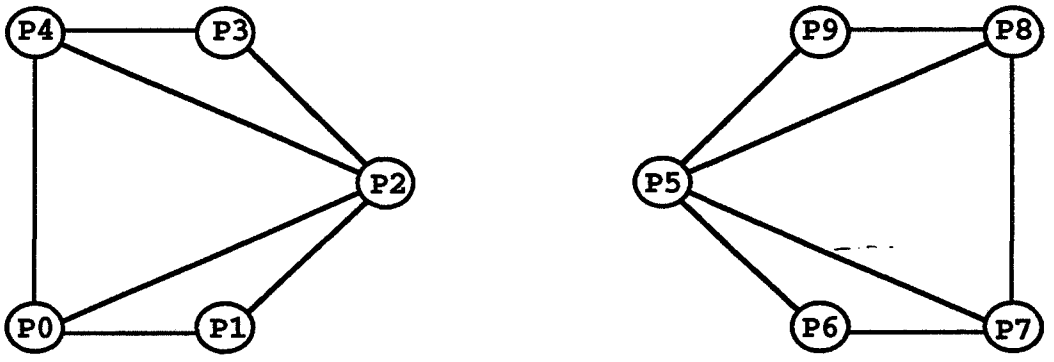


Figure 36 : Graphe G.

Les arêtes P3P9, P2P5 et P1P6 sont les trois arêtes d'articulation. Le graphe G sera décomposé pour donner les composantes G1 et G2 suivantes. G1 et G2 sont 1-rigide.



Graphe G1. Graphe G2.

Figure 37.

ETAPE 4 :

Actuellement, si aucune des trois étapes précédentes n'est applicable, notre programme procède à la résolution du système de contraintes par la méthode numérique de Bissection. Cette méthode permet de retrouver toutes les figures géométriques satisfaisant un ensemble de contraintes dans un domaine donné (voir chapitre 1).

Cependant, en théorie, on peut considérer, au delà des graphes 1-rigides ou 2-rigides, des graphes 3-rigides, 4-rigides, etc ... . Ce point est abordé en section six.

4.4. CAS DES SOLUTIONS MULTIPLES

4.4.1. NOMBRE DE SOLUTIONS

Le nombre de solutions d'un ensemble de contraintes correspondant à un graphe rigide est en général une fonction exponentielle du nombre d'entités géométriques utilisées. Ceci résulte du fait que les règles utilisées pour déterminer une entité géométrique peuvent donner de une à huit solutions différentes pour chaque entité (par exemple un cercle tangent à trois cercles peut avoir huit solutions différentes).

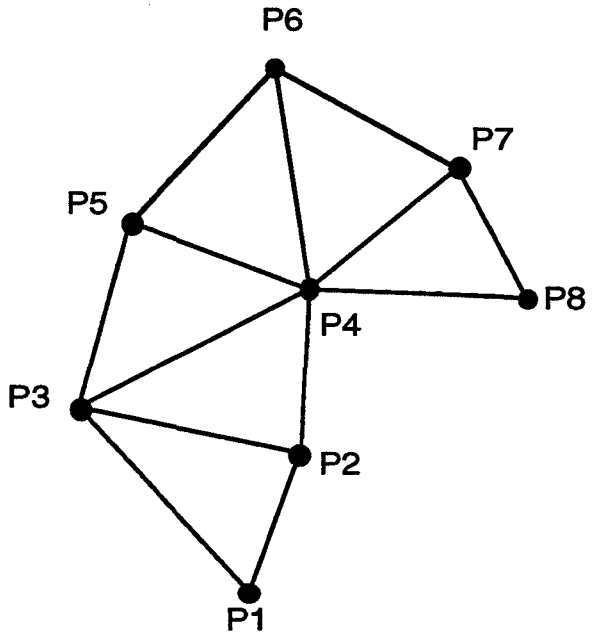


Figure 38.

Considérons un problème de contraintes comprenant n points et 2n - 3 contraintes de distances entre ces points. Nous pouvons avoir 2^{n-2} solutions différentes pour ce problème : en effet, si nous fixons deux points, nous aurons en général pour tous les autres points (intersection de deux cercles) deux solutions possibles. La figure 38 illustre un tel problème de contraintes. Ce problème possède 2^6 solutions différentes.

4.4.2. CHOIX DE LA SOLUTION

Donner toutes les solutions d'un problème de contraintes peut nécessiter beaucoup de temps de calcul si nous avons un nombre exponentiel de solutions.

Pour notre part, la construction de la figure solution se fait de façon incrémentale. A chaque étape de la résolution, pour déterminer une entité géométrique ayant plusieurs solutions, nous effectuons une permutation circulaire sur la liste de ces solutions. Nous donnons une solution que nous visualisons à l'écran (c'est l'entité géométrique qui est visualisée) et si l'utilisateur ne l'accepte pas, nous donnons une autre solution jusqu'à ce que l'utilisateur en confirme une.

Du fait de la méthode utilisée pour le choix de la solution, la résolution se fait de façon interactive sans nécessiter beaucoup de temps de calcul.

5. RESULTATS

5.1. IMPLEMENTATION

Les algorithmes décrits dans ce chapitre ont été implantés en langage LeLisp et SMECI sur une station Sun, sous le système Unix. AIDA est un système de construction d'interfaces graphiques sophistiquées pour LeLisp. LeLisp, AIDA et SMECI sont des produits de la société ILOG.

5.2. EXEMPLES

Nous présentons dans ce qui suit quelques exemples de problèmes de contraintes résolus par le système développé. La résolution est interactive.

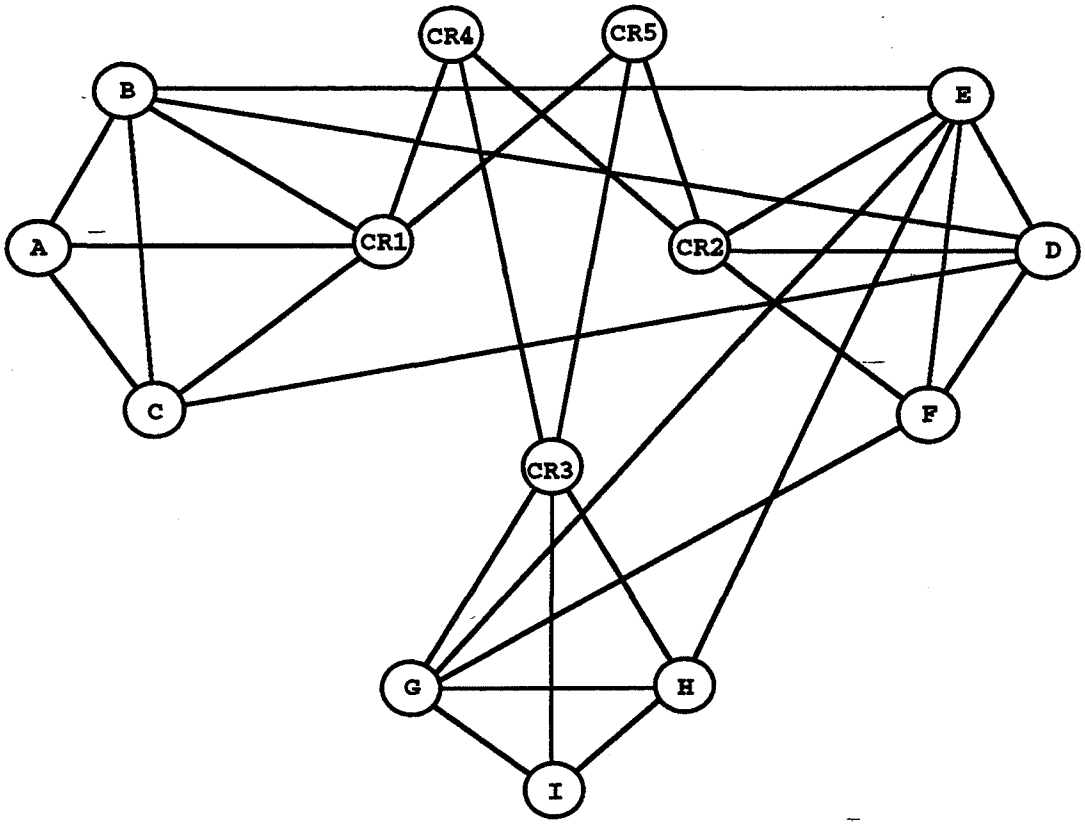


Figure 39.

Fig. 40 Les entités géométriques considérées sont les points A, B, C, D, E, F, G, H et I et les cercles CR1, CR2, CR3, CR4 et CR5. Les arêtes entre les points représentent des contraintes de distance ; le cercle CR1 passe par les points A, B et C ; le cercle CR2 passe par les points D, E et F ; le cercle CR3 passe par les points H, I et J ; les cercles CR4 et CR5 sont tangents aux cercles CR1, CR2 et CR3. Le graphe des contraintes correspondant est donné par la figure 39. La résolution est faite par l'étape une.

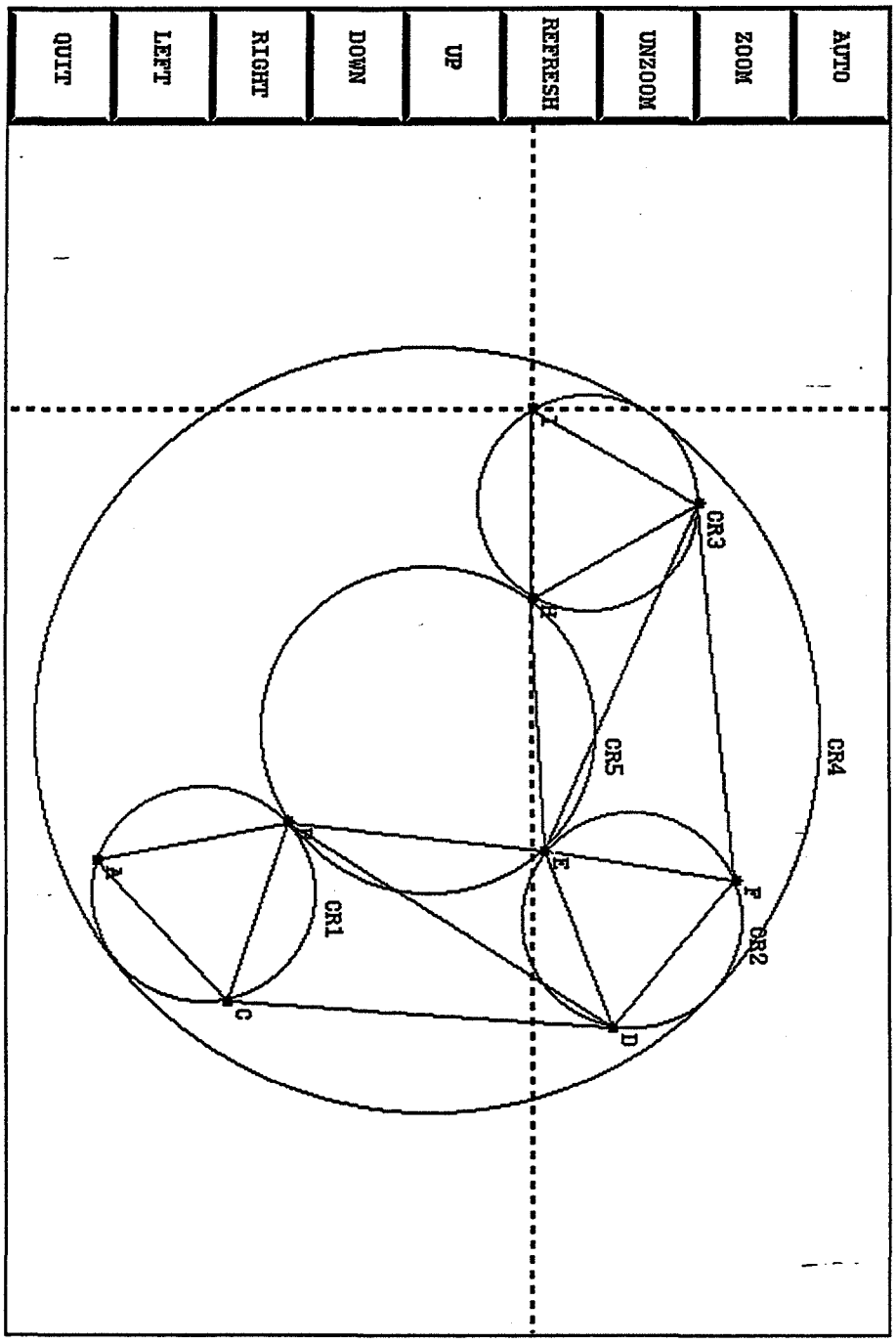

QUIT
LEFT
RIGHT
DOWN
UP
REFRESH
UNZOOM
ZOOM
AUTO
A
B
C
D
E
F
CR1
CR2
CR3
CR4
CR5

Figure 40.

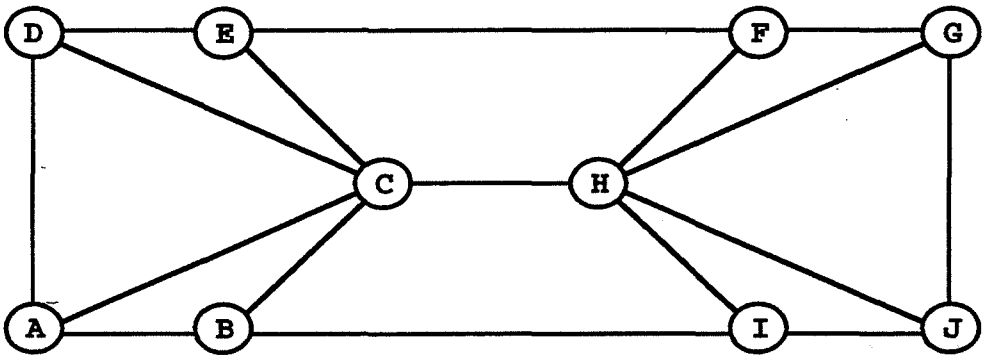


Figure 41.

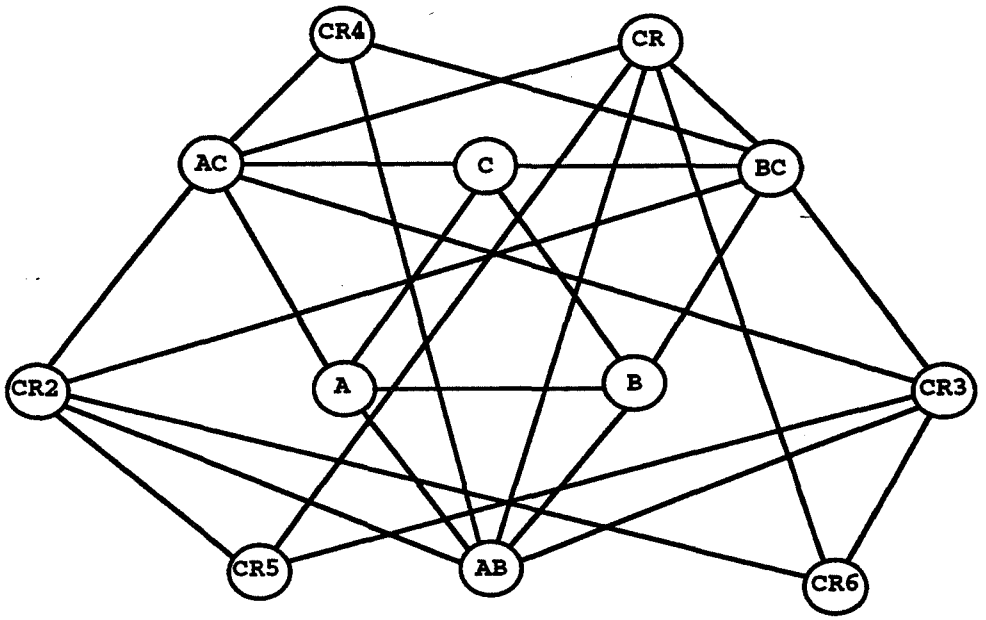


Figure 42.

Fig. 43 Les entités géométriques considérées sont les points A, B, C, D, E, F, G, H et I et les arêtes entre ces points représentent des contraintes de distance. Le graphe de contraintes est donné par la figure 41. La résolution est faite par l'étape une et l'étape trois.

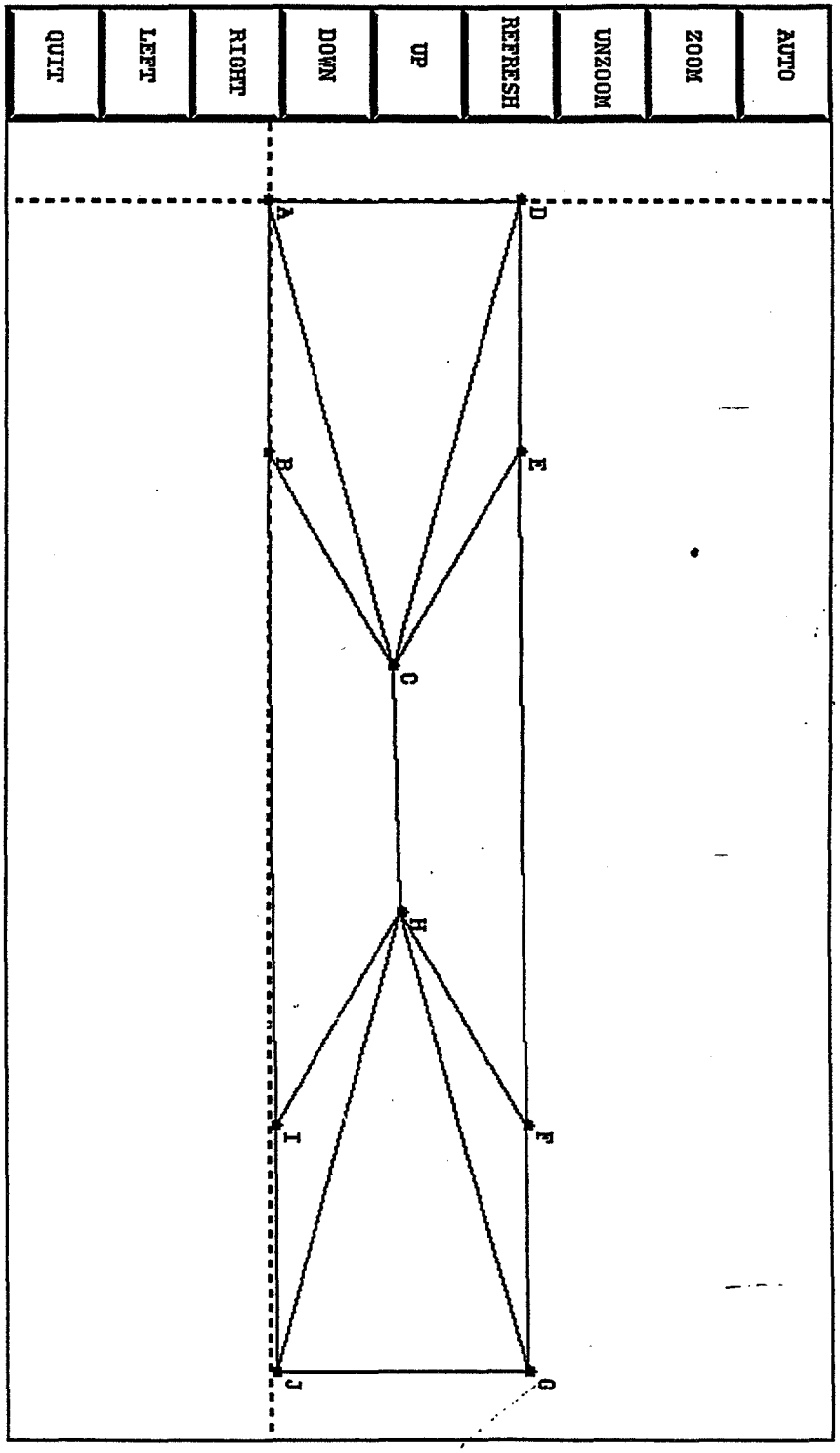

QUIT
LEFT
RIGHT
DOWN
UP
REFRESH
UNZOOM
ZOOM
AUTO
A
B
C
D
E
F
G
H
I
J

Figure 43.

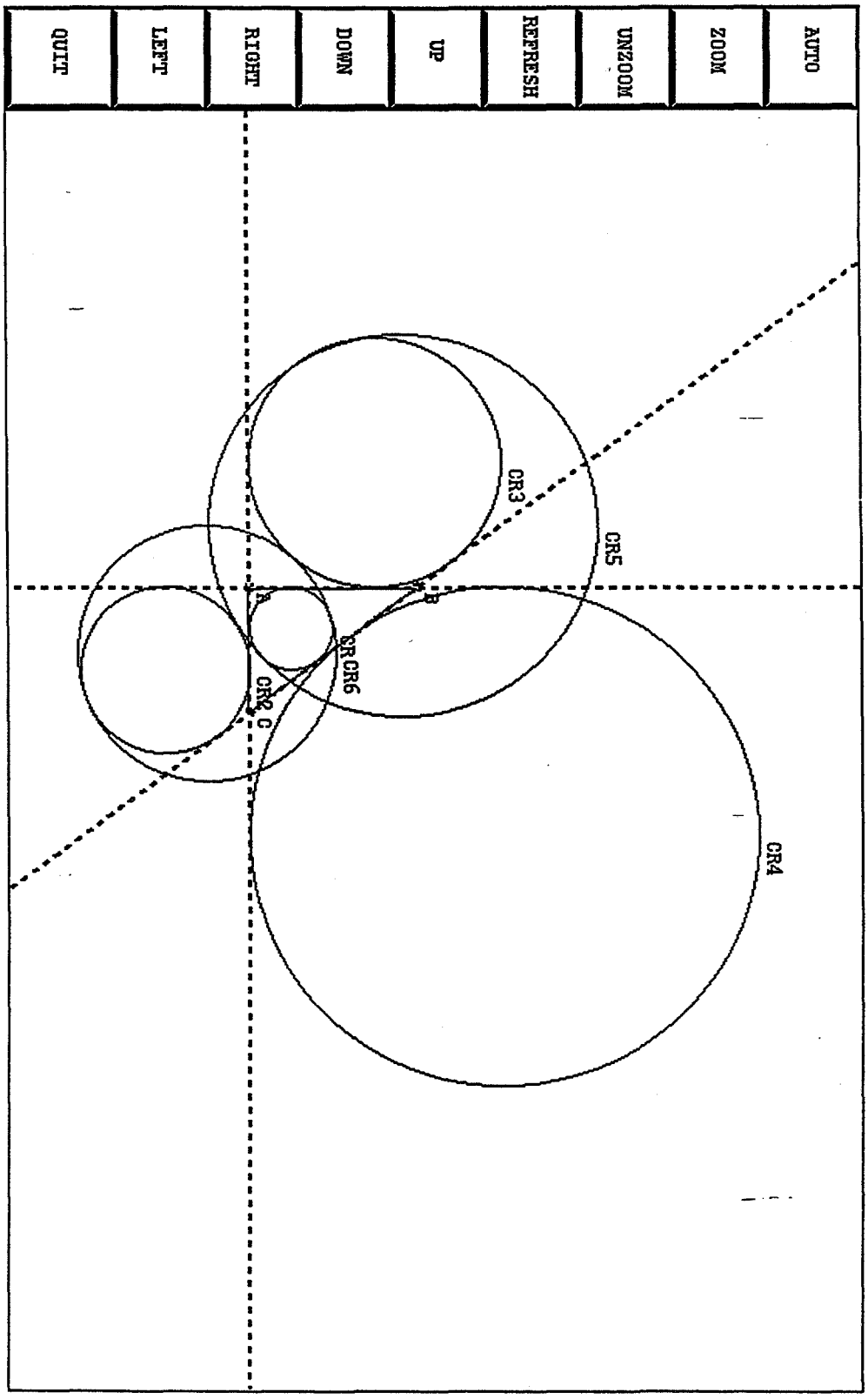

QUIT
LEFT
RIGHT
DOWN
UP
REFRESH
UNZOOM
ZOOM
AUTO
CR3
CR5
CR6
CR2
CR4

Figure 44.

Fig. 44 Les entités géométriques considérées sont les points A, B et C, les cercles CR, CR2, CR3, CR4, CR5 et CR6 et les droites AB, AC et BC. Les points A, B et C sont liés par des contraintes de distance ; les cercle CR, CR2, CR3 et CR4 sont tangents aux droites AB, AC et BC ; les cercle CR5 et CR6 sont tangents aux cercles CR, CR2 et CR3. Le graphe de contraintes est donné par la figure 42. La résolution est faite par l'étape une.

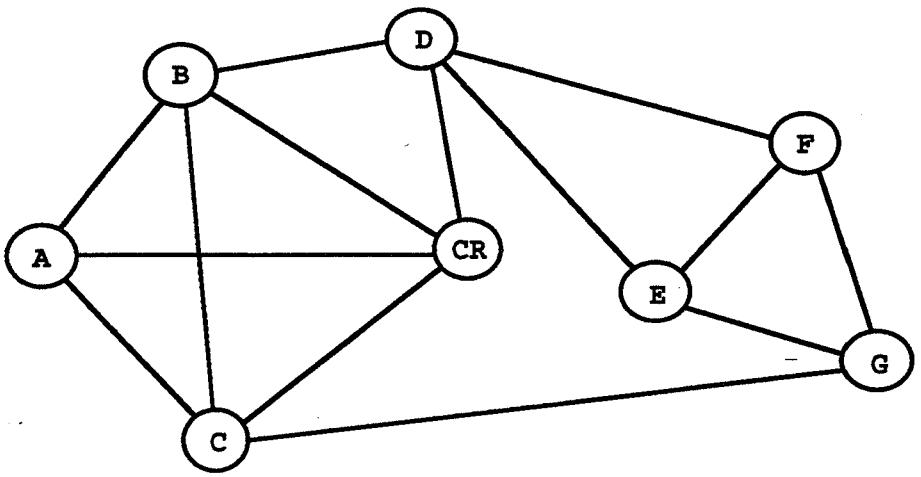


Figure 45.

Fig. 46 Les entités géométriques considérées sont les points A, B, C, D, E, F et G et le cercles CR. Les arêtes entre les points représentent des contraintes de distance ; le cercle CR passe par les points A, B et C. Le graphe de contraintes est donné par la figure 45. La résolution est faite par l'étape une et l'étape 2.

Fig. 47 Même graphe de contraintes que figure 46.

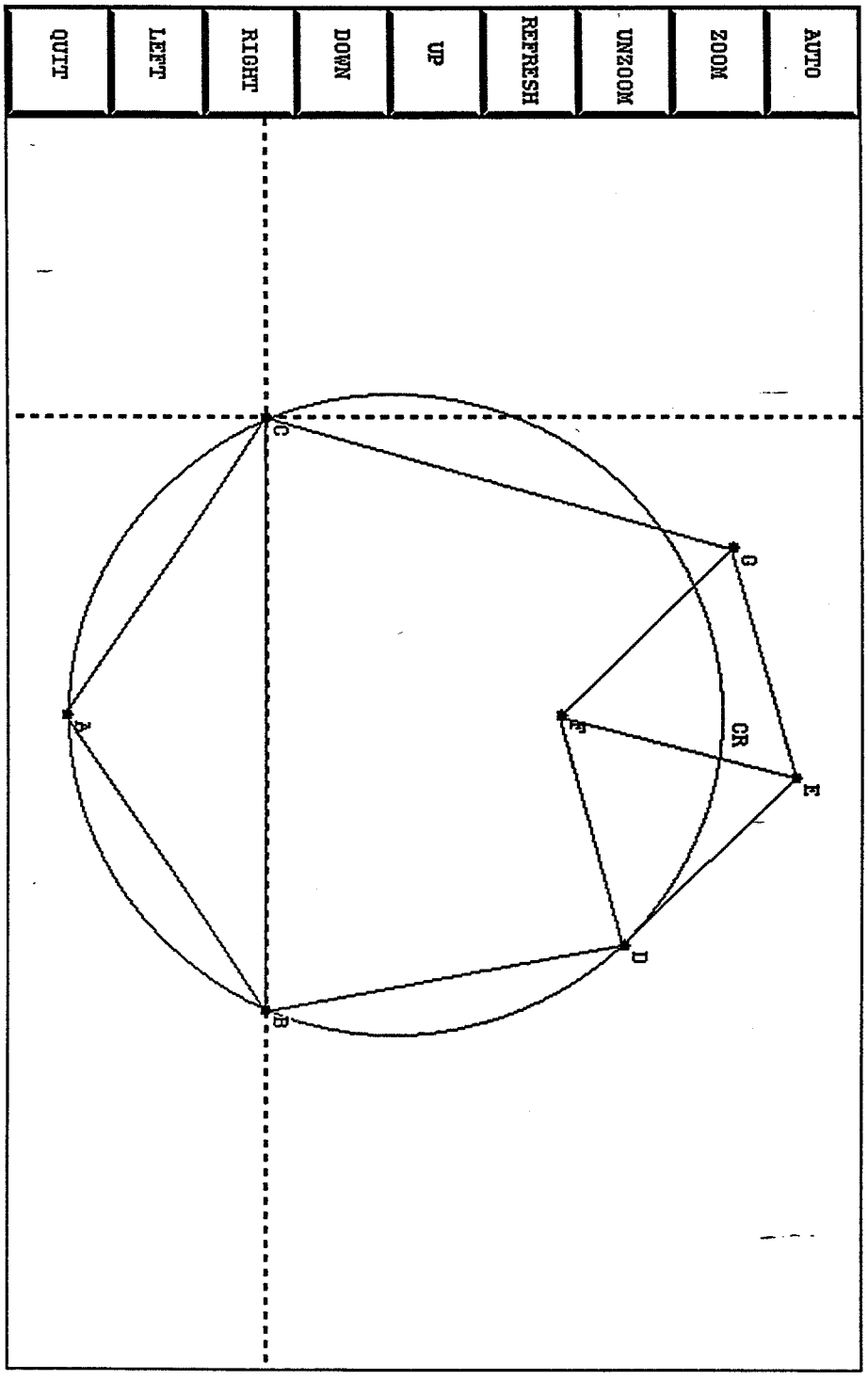


Figure 46.

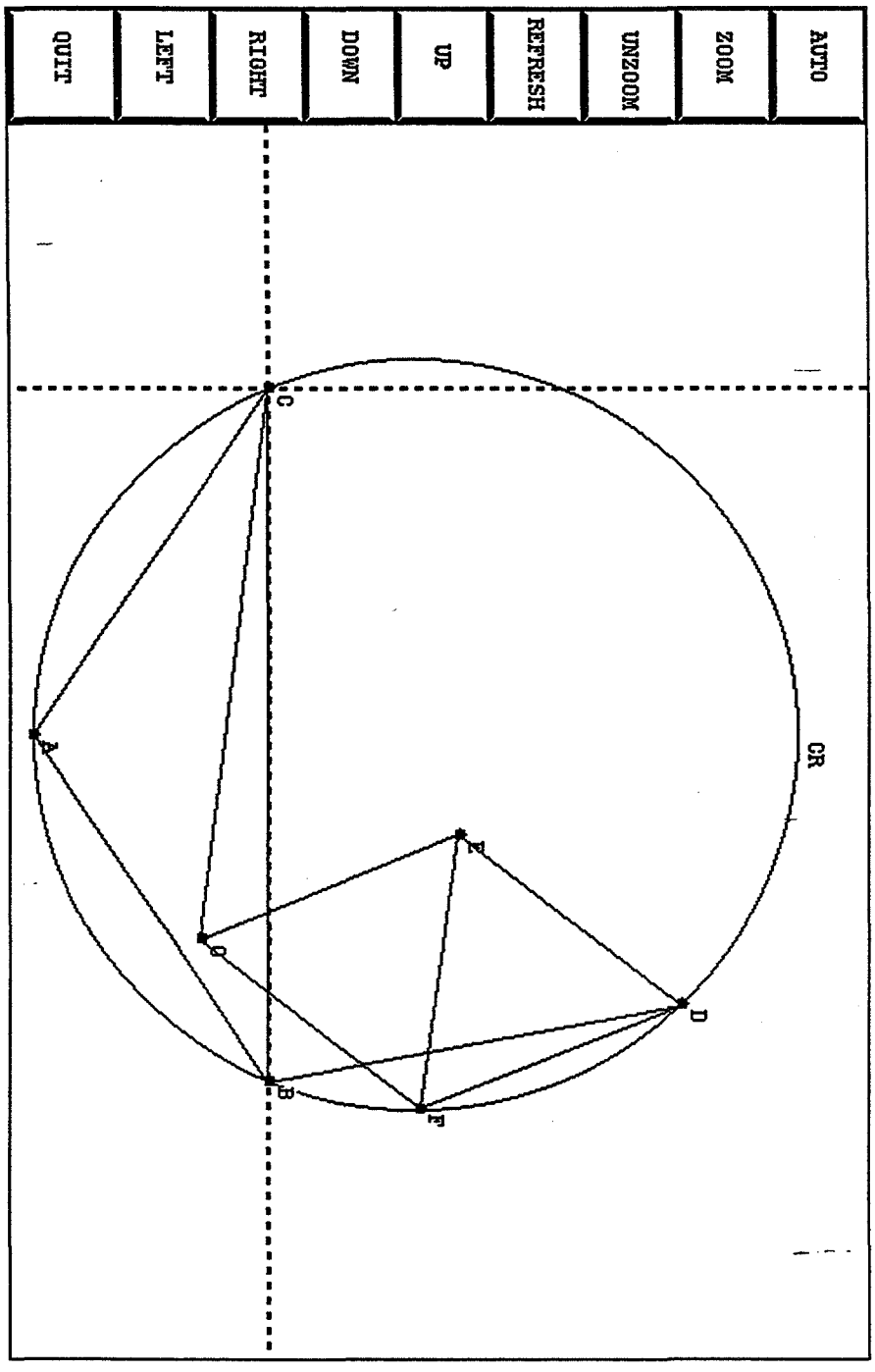


Figure 47.

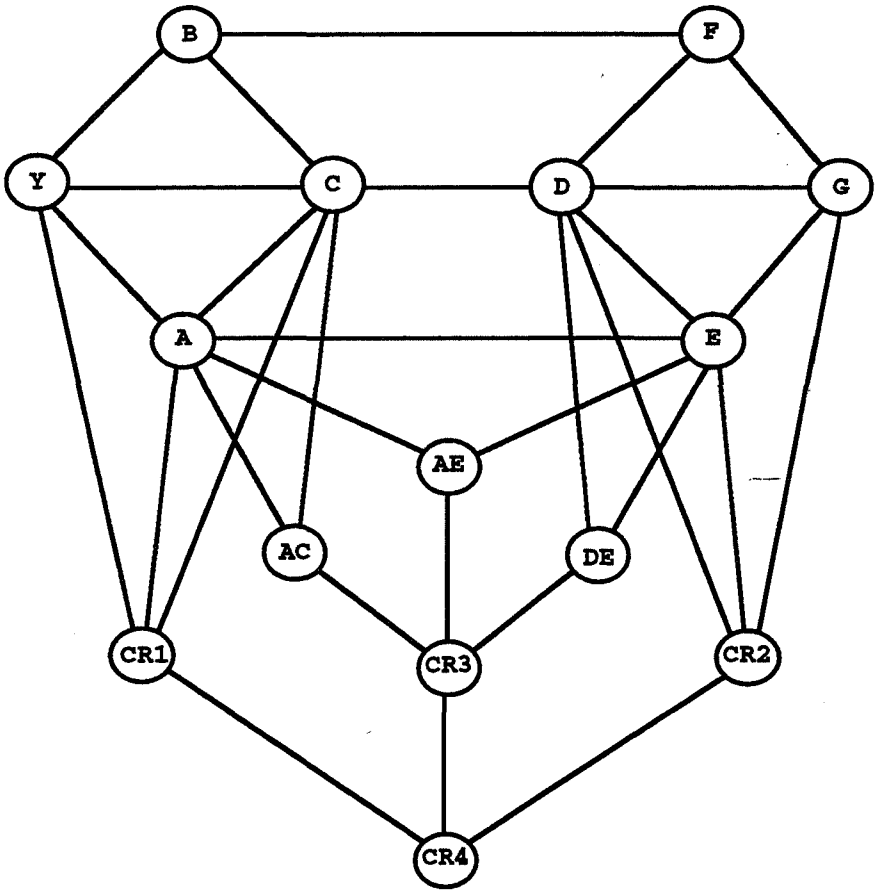


Figure 48.

Fig. 49 Les entités géométriques considérées sont les points A, B, C, D, E, F, G et Y, les cercles CR1, CR2, CR3 et CR4 et les droites AC, DE et AF. Les arêtes entre les points représentent des contraintes de distance ; le cercle CR1 passe par les points A, C et Y ; le cercle CR2 passe par les points D, E et G ; le cercle CR3 est tangent aux droites AC, DE et AF ; le cercle CR4 est tangent aux trois cercles CR1, CR2 et CR3. Le graphe de contraintes est donné par la figure 48. La résolution est faite par l'étape une et l'étape trois.

Fig. 50 Les entités géométriques considérées sont des points et les arêtes entre ces points représentent des contraintes de distance. La résolution est faite par l'étape une.

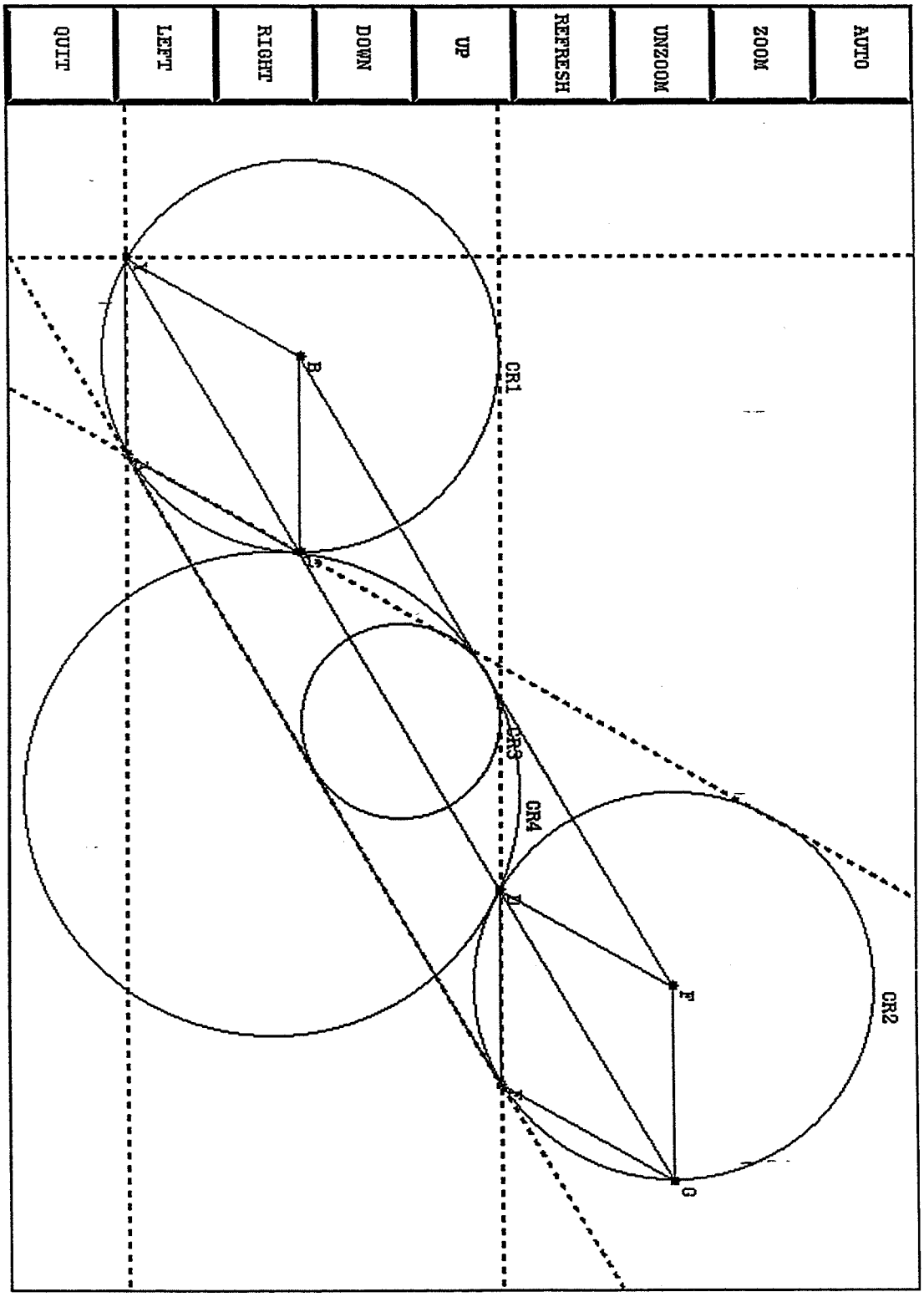

QUIT
LEFT
RIGHT
DOWN
UP
REFRESH
UNZOOM
ZOOM
AUTO
B
CR1
CR3
CR4
F
CR2
G

Figure 49.

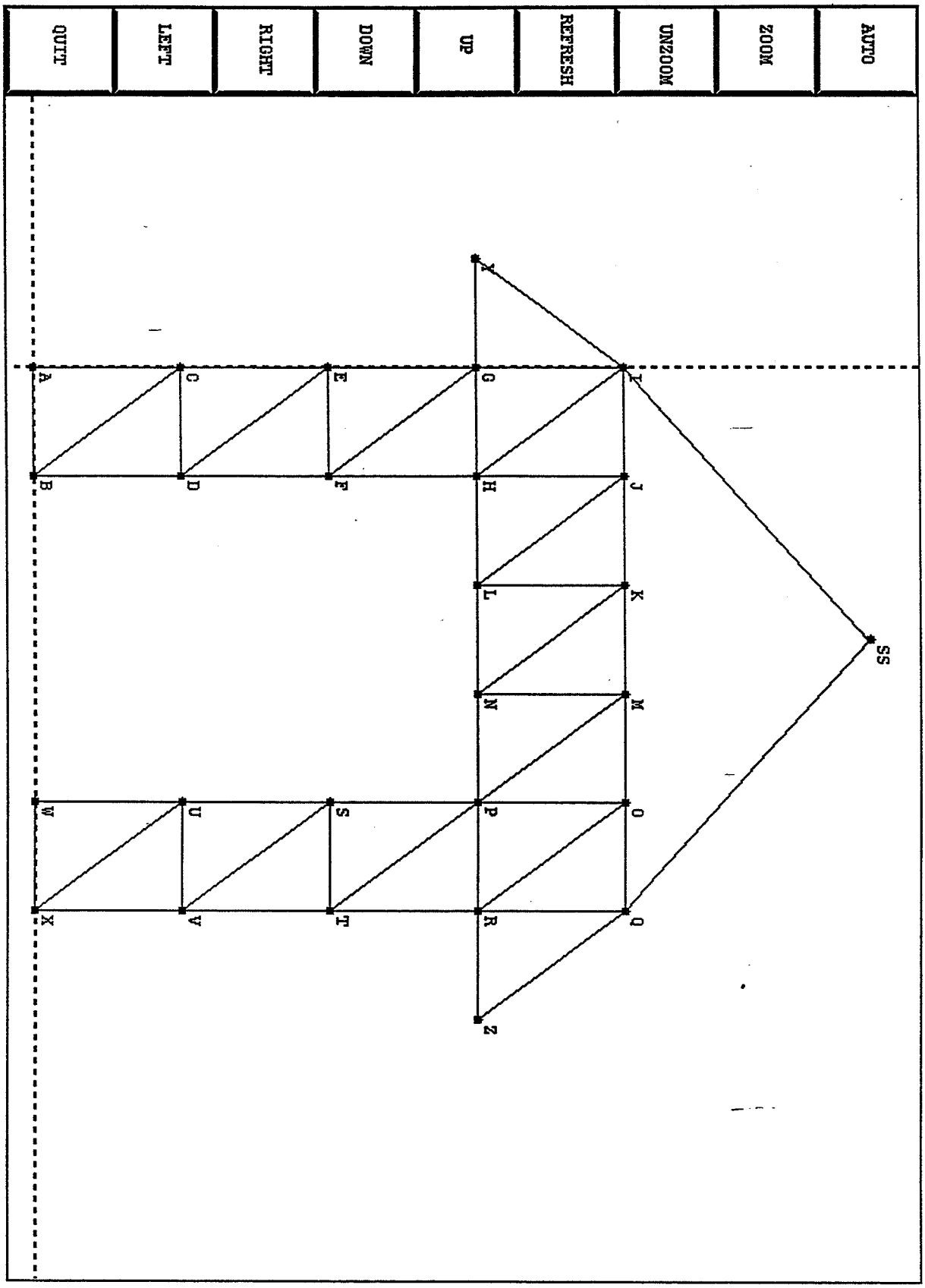

QUIT
LEFT
RIGHT
DOWN
UP
REFRESH
UNZOOM
ZOOM
AUTO
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z
SS

Figure 50.

6. DETECTION DE TOUS LES SOUS-RIGIDES

Les algorithmes donnés précédemment procédaient à la réduction du graphe de contraintes par la recherche d'ensembles d'articulation. Une deuxième façon de procéder est de parcourir les arêtes du graphe de contraintes, fixer les deux sommets extrémités de chaque arête et essayer de déduire d'autres entités géométriques du graphe par propagation.

6.1. ALGORITHME

Dans les sections précédentes, nous avons donné des algorithmes de résolution de problèmes de contraintes ayant un graphe 1-rigide ou 2-rigide. Nous proposons, dans ce qui suit, un algorithme linéaire détectant et résolvant tous les graphes 1-rigides d'un graphe de contrainte G rigide quelconque dans l'espace bi-dimensionnel.

Le principe de cet algorithme est le suivant : Les arêtes du graphe sont parcourues une à une ; si une arête appartient déjà à sous-graphe 1-rigide, cette arête est ignorée ; si une arête n'appartient à aucun sous-graphe 1-rigide déjà détecté, cette arête est fixée et ses deux sommets extrémités sont stockés dans un nouveau rigide ; chaque sommet se rattachant à cet ensemble par un nombre d'arêtes égal à son degré de liberté y est ajouté : nous obtenons ainsi les sommets du sous-graphe 1-rigide. Cette opération est réitérée jusqu'à ce qu'aucun sommet ne puisse plus être ajouté ; si le cardinal de cet ensemble est supérieur à deux, un nouveau sous-graphe 1-rigide a été détecté, sinon l'ensemble considéré ne contient que l'arête de départ qui est alors ignorée.

Algorithme :

```
m := nombre d'arêtes du graphe G;
n := nombre de sommets du graphe G;
i := 0;
j := 1;
fin1 := faux;
/* Les ensembles Li contiendront les sommets des sous-graphes 1-rigide */;
```

Tant que ($j \leq m$) et (non fin1)
Faire
Si l'arête $a_j \notin L_k$ ($k = 1$ à i) /* $a_j = (A, B)$ */
Alors
fixer a_j ; /* en fait, nous supposons que l'arête est fixée ; nous n'affectons pas de coordonnées aux deux sommets extrémités de cette arête */
$i := i + 1$;
$L_i := \{A, B\}$;
fin2 := faux;
Tant que non fin2
Faire
LTMP := L_i;
k := 0;
Tant que LTMP non vide
Faire
S := premier élément de la liste LTMP;
LTMP := LTMP - {S};
k := k + 1;
LV_k := voisins (S) - L_i;
Fait;

ENSP := {ensemble de sommets de type point ou droite apparaissant deux fois dans les listes LV_k};
ENSC := {ensemble de sommets de type cercle apparaissant trois fois dans les listes LV_k};
ENS := ENSP $\cup$ ENSC;
Si ENS = $\emptyset$;
Alors fin2 := vrai;
Sinon $L_i := L_i \cup$ ENS;
fsi;
Fait;

Cas où
• cardinal (L_i) = 2 **Alors** i := i - 1;
/* L'ensemble L_i ne contient qu'une arête */;

• $\exists$ k = 1 à (i - 1) tel que (cardinal ($L_i \cap L_k$)) > 2

Alors

$L_k := L_k \cup^* L_i$; i := i - 1;

/* l'opérateur $\cup^*$ désigne l'union sans doublons */;

- cardinal ($\cup L_k$, k = 1 à i) = n **Alors** fin1 := vrai;

fincas;

fsi;

j := j + 1;

Fait;

Les ensembles L_k (k = 1 à i) contiennent les sommets des graphes 1-rigides.

Exemple :

Trois sous-graphes 1-rigides (en l'occurrence des triangles) sont détectés dans le graphe de la figure 51.

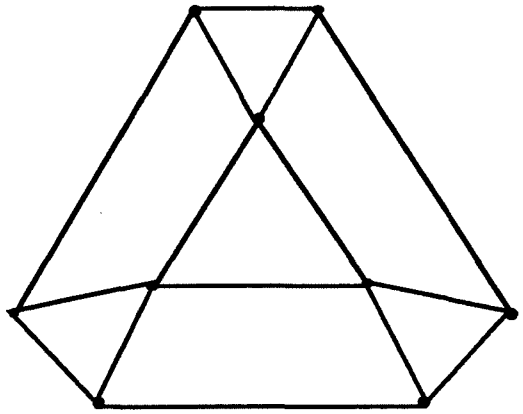


Figure 51.

6.2. GRAPHE 3-RIGIDE

Définition :

Un graphe de contraintes G rigide sera dit 3-rigide si il est composé de trois composantes rigides reliées entre elles, deux à deux, par des ensembles de deux arêtes (i.e contraintes).

Un exemple de graphe 3-rigide est donné par la figure 52. Les graphes Gi sont des graphes rigides.

La résolution d'un graphe 3-rigide débute par la résolution des sous-graphes rigides le composant. La deuxième étape consiste à fixer une composante (par exemple, celle ayant le plus grand nombre d'entités géométriques) et replacer les deux autres composantes rigides à l'aide de rotations et translations. Le déplacement (rotation et translation) à appliquer à une composante induit trois inconnues. Cette deuxième étape revient en fait à résoudre un système de six équations à six inconnues.

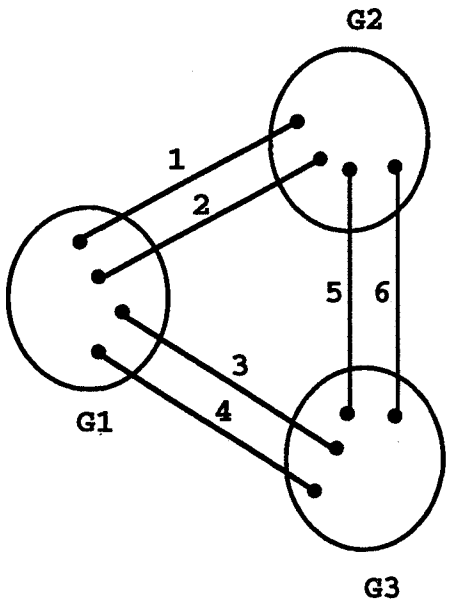


Figure 52.

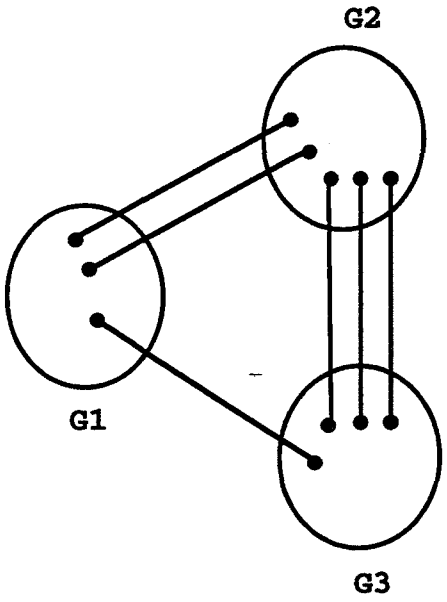


Figure 53.

Remarque :

On ne considère pas comme 3-rigide un graphe tel celui illustré par la figure 53. Bien qu'étant l'assemblage de trois rigides, ce graphe est une composition de graphes 2-rigides ; G2 ∪ G3 est un 2-rigide et G1 ∪ (G2 ∪ G3) est aussi un 2-rigide.

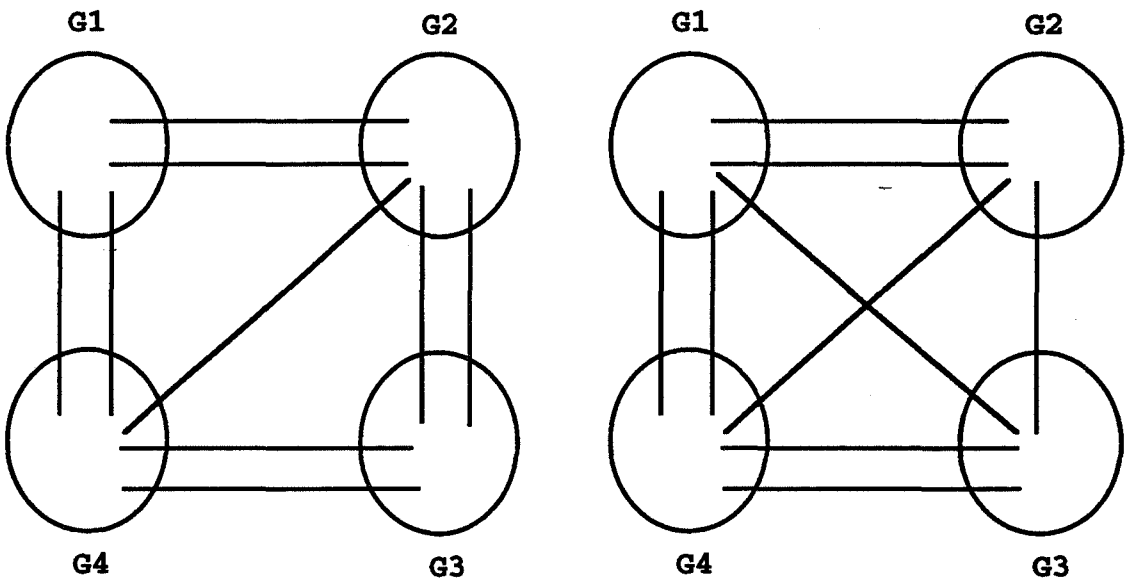
6.3. GRAPHE 4-RIGIDE

Définition :

Un graphe de contraintes G rigide sera dit 4-rigide si il est composé de quatre composantes rigides reliées entre elles par des ensembles d'arêtes (i.e contraintes) et aucun des triplets de G n'est un 3-rigide.

Comme pour un graphe 3-rigide, la résolution d'un graphe 4-rigide débute par la résolution des sous-graphes rigides le composant. La deuxième étape consiste à fixer une composante et replacer les trois autres composantes rigides à l'aide de rotations et translations. Cette deuxième étape revient en fait à résoudre un système de neuf équations à neuf inconnues.

Des exemples de graphes 4-rigides sont donnés par les figures 54.a. et 54.b. Les graphes Gi sont des graphes rigides.



(a) (b)

Figure 54.

Remarque :

Aucun des trois triplets d'un graphe 4-rigide n'est un 3-rigide.

7. CONCLUSION

Le système que nous avons développé permet "d'optimiser" d'une certaine façon la résolution de différentes configurations induites par des contraintes de distances, d'angles et de tangences entre points, droites et cercles de rayon connu ou non. Cette optimisation consiste à décomposer le problème de contraintes en problèmes géométriques élémentaires et à combiner les solutions. Cette résolution géométrique est, dans ce cas, plus efficace que les méthodes algébriques. L'utilisateur peut, en outre, avoir les détails du processus de résolution.

Cependant, on ne peut résoudre géométriquement tous les systèmes de contraintes. La méthode "purement" géométrique est moins générale que les méthodes algébriques. Il existe des problèmes de contraintes où la résolution ne peut se faire que globalement d'où le recours à une méthode numérique dans ces cas là. Un exemple de graphe où la résolution ne peut se faire que globalement est donné par la figure 55.

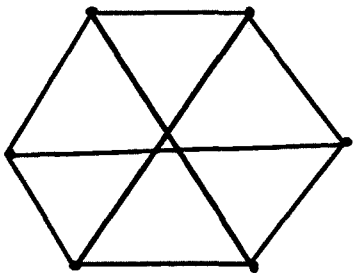


Figure 55.

Un autre problème de cette méthode est le nombre de cas à traiter (règles à implémenter) qui augmente fortement si on a un très grand nombre de types de contraintes géométriques. De plus, l'ajout d'un nouveau type de contraintes n'est pas simple.

Une amélioration qui peut être apportée au programme développé est de trouver efficacement les arêtes d'articulations d'un graphe k-rigide pour pouvoir résoudre géométriquement ce graphe.

CONCLUSION

Dans cette thèse, nous avons fait un survol sur les différentes approches de modélisation géométrique par contraintes et nous avons proposé deux méthodes de résolution du système de contraintes.

La première partie est consacré à l'étude des approches de modélisation des contraintes. Nous avons présenté un ensemble de méthodes pour résoudre les contraintes. Ces méthodes sont classifiées en deux catégories : les approches algébriques et les approches géométriques. Ces approches sont analysées à travers les travaux de différents auteurs. Ce sont des techniques classiques de résolution. Nous avons aussi mis en exergue les principaux avantages et inconvénients de chaque approche de résolution. La détermination des coordonnées de sommets à partir des distances entre sommets est étudiée en s'appuyant sur la théorie des graphes. Cette partie constitue un état de l'art des méthodes de résolution.

Dans la seconde partie nous avons présenté deux méthodes de résolution du système de contraintes. La première méthode est basée sur les graphes bipartis sous-jacents aux systèmes d'équations correspondant aux contraintes. Nous avons montré qu'il etait possible de décomposer polynomialement ces systèmes en sous-systèmes sur-contraints (plus d'équations que d'inconnues), sous-contraints (plus d'inconnues que d'équations) et bien-contraints (autant d'équations que d'inconnues) à partir du graphe biparti. Cette décomposition constitue une première aide à l'utilisateur car elle permet un débogage du système d'équations. Nous avons introduit aussi une procédure efficace de décomposition des systèmes bien-contraints en sous-systèmes irréductibles c.à.d ne pouvant plus être décomposés. Ces décompositions accélérent grandement la résolution dans le cas des grands systèmes réductibles. Nous pouvons, en outre, utiliser cette méthode pour tester la rigidité de structures bidimensionnelles.

La deuxième méthode concerne le développement d'un système de résolution de contraintes bi-dimensionnelles par une méthode géométrique. Comme nous l'avons vu dans le tour d'horizon, cette méthode exploite "la connaissance géométrique" des contraintes. Nous avons étudié les différentes configurations induites par des contraintes de distances, d'angles et de tangences entre points, droites et cercles de rayon connu ou non. Nous avons proposé des algorithmes permettant de décomposer un problème de contraintes en problèmes élémentaires à résoudre et de combiner les solutions. Les entités géométriques sont déterminées par un algorithme de réduction de graphes et un système à base de règles. Lorsque aucune déduction géométrique ne peut être faite, nous avons utilisé une méthode numérique (bissection) pour résoudre le système de contraintes. La figure solution est donnée à un déplacement près.

Cependant, le domaine est encore vaste et de nombreuses améliorations peuvent être apportées. Les méthodes que nous avons proposées sont opérationnelles pour quelques dizaines d'entités géométriques. Le test du système développé sur des milliers, voir plus, d'entités géométriques reste à effectuer pour savoir si l'efficacité de ce système n'est pas altérée. Une extension possible du système développé est d'introduire d'autres types de contraintes, telles que les contraintes mécaniques ou physiques. Le cas 3D, dans le cas d'une résolution "purement" géométrique, est encore un domaine de recherche. En effet, comme nous l'avons vu, il n'existe pas de caractérisation de rigidité de structures tridimensionnelles en termes de graphes et les logiciels développés se limitent à des exemples simples.

ANNEXE

L'ensemble de règles utilisées dans notre base est le suivant :

- Règles permettant de déduire un point :
  - point appartenant à une droite et un cercle.
  - point appartenant à deux cercles.
  - point appartenant à deux droites.

- Règles permettant de déduire une droite :
  - droite passant par deux points.
  - droite passant par un point et faisant un angle avec une droite.
  - droite passant par un point et parallèle à une droite.
  - droite passant par un point et tangente à un cercle.
  - droite tangente à deux cercles.
  - droite tangente à un cercle et parallèle à une droite.
  - droite tangente à un cercle et faisant un angle avec une droite.

- Règles permettant de déduire un cercle :
  - cercle tangent à trois droites.
  - cercle tangent à un cercle et deux droites.
  - cercle passant par un point et tangent à deux droites.
  - cercle tangent à deux cercles et une droite.
  - cercle tangent à un cercle et une droite et passant par un point.
  - cercle tangent à une droite et passant par deux points.
  - cercle tangent à trois cercles.
  - cercle tangent à deux cercles et passant par un point.
  - cercle tangent à un cercle et passant par deux points.
  - cercle passant par trois points.

BIBLIOGRAPHIE

[AJM 93] S. Ait-Aoudia, R. Jegou et D. Michelucci. *Reduction of constraint systems.* COMPUGRAPHICS '93 Third International Conference on Computational Graphics and Visualisation Techniques, Algarve Portugal. Dec 93. pp 331-340.

[Ait 92] S. Ait-Aoudia. *Modélisation 3D à l'aide de contraintes géométriques.* GROPLAN '92, Nantes France, Nov. 92. pp 47-54.

[Ait 94] S. Ait-Aoudia. *Solveur géométrique de contraintes.* Journées 3IA '94, Infographie Interactive et Intelligence Artificielle, Limoges France, Avril 94. pp 145-165.

[Alde 88] B. Aldefeld. *Variation of geometries based on a geometric reasoning method.* CAD vol.20, num.3, April 88. pp 117-126.

[Berg 57] C. Berge. *Two theorems in graph theory.* Proc. Nat. Ac. Sciences, USA, 43, p. 842, 1957.

[Berg 83] C. Berge. *Graphes.* Gauthier-Villars, Bordas 1983.

[Barf 87] L.A. Barford. *A graphical language-based editor for generic solid models represented by constraints.* PhD Thesis, Cornell University, May 1987.

[BD 86] A. Borning et R. Duisberg. *Constraint-based tools for building user interfaces.* ACM Transactions on Graphics, vol. 5, num. 4, Oct. 86. pp 345-374.

[Born 79] A. Borning. *THINGLAB: a constraint-oriented simulation laboratory.* Ph.D Thesis, Stanford University 1979.

[BP 93] S.A. Buchanan et A. Pennington. *Constraint definition system : a computer-algebra based approach to solving geometric constraint problems.* CAD, vol. 25, num. 12, Dec. 93. pp 741-750.

[Brau 88] G. Braun. *Sur la programmation de constructions géométriques.* Thèse de doctorat, Université Louis Pasteur, Strasbourg, juin 1988.

[Brau 89] G. Braun. *Sur la programmation de constructions géométriques.* PIXIM '89, Sept. 89. pp 419-433.

[Brüd 86] B. Brüderlin. *Constructing three-dimensional geometric objects defined by constraints.* Proceedings 1986. Workshop on Interactive 3D Graphics. pp. 111-129, Chapel Hill, N.C, Oct 86.

[Brüd 88] B. Brüderlin. *Automatizing geometric proofs and constructions.* Computational Geometry and its Applications. pp 233-252. Lecture notes in Computer Science, 333, Springer Verlag, Mars 88.

[Buch 65] B. Buchberger. *An algorithm for finding a basis for the residue class ring of a zero-dimensional polynomial ideal.* Ph.D Thesis, Innsbruck University, 1965.

[Buch 70] B. Buchberger. *An algorithmical criterion for the solvability of algebraic systems of equations.* Aequations Mathematicae 4, fasc. 3, 1970. pp 374-383.

[Buch 79] B. Buchberger. *A criterion for detecting unnecessary reductions in the construction of Gröbner bases.* Proc. EUROSAM'79, Marseille, June 79.

[Buch 85] B. Buchberger. *Basic features and development of the critical-pair / completion procedure.* Proc. Rewriting Techniques and Applications, Dijon, May 85.

[Buth 79] M. Buthion. *Un programme qui résout formellement des problèmes de constructions géométriques.* R.A.I.R.O.. Informatique / Computer Science, vol. 13, num. 1, 1979. pp 73-106.

[Carr 89] J.C. Carrega. *Théorie des corps, la règle et le compas.* Herman, 1989.

[CDG 85] U. Cugini, C. Devoti et P. Galli. *System for parametric definition of engineering drawings.* Micad '85.

[CFV 88] U. Cugini, F. Folini et I. Vicini. *A procedural system for the definition and storage of technical drawings in parametric form.* Eurographics'88, Amsterdam, The Nederlands. pp 183-196.

[Chyz 85] W. Chyz. *Constraint management for CSG.* Master thesis, MIT, June 1985.

[Ciar 82] P.G. Ciarlet. *Introduction à l'analyse numérique matricielle et à l'optimisation.* Masson, 1982.

[CS 88] J.C.H. Chung et M.D. Schussel. *Comparison of variational and parametric design.* Revue de CFAO et d'Infographie, vol. 4, num. 4, 1988. pp 81-101.

[CSY 87] S.C. Chou, W.F. Schelter et J.G. Yang. *Characteristic sets and Gröbner bases in geometry theorem proving.* Workshop on Computer-Aided Geometric Reasoning, INRIA, Sophia Antipolis, June 87. pp 29-56.

[DB 74] G. Dahlquist et A . Bjorck. *Numerical Methods.* Prentice Hall series in automatic computation, 1974.

[DM 58] A.L. Dulmage, N.S. Mendelsohn. *Coverings of bipartite graphs.* Canad. J. Math., 10, 517-534, 1958.

[DM 59] A.L. Dulmage, N.S. Mendelsohn. *A structure theory of bipartite graphs of finite exterior dimension.* Trans. of Royal Soc. Canada, Section III, 53, 1-13, 1959.

[DM 62] A.L. Dulmage, N.S. Mendelsohn. *On the inversion of sparsed matrices.* Math. Comput., 16, 494-496, 1962.

[DM 63] A.L. Dulmage, N.S. Mendelsohn. *Two algorithms for bipartite graphs.* SIAM J., 11, 183-194, 1963.

[DM 93] A.K. Dhingra et N.K. Mani. *A symbolic computing approach.* CAD, vol. 25, num. 5, May 93. pp 300-310.

[Duff 77] I.S. Duff. *A Survey of sparse matrix research.* Proceedings of the IEEE, vol. 65, no 4, 500-535, april 1977.

[Duis 86] R. Duisberg. *Constraint based animation. The implementation of temporal constraints.* Ph.D Thesis, University of Washington 1986.

[DY 93] J.P. Dedieu et J.C. Yacoubsohn. *The BissEx and newBissEx procedures : two seminumerial algorithms for solving polynomial systems.* SEA '93, Marseille 1993. pp 1-27.

[EY 88] L.W. Ericson et C.K. Yap. *The design of LINETOOL a geometric editor.* Fourth Annual Symposium on Computational Geometry, ACM Press, June 88. pp 83-92.

[GGJZ 92a] H. Gaoua, Y. Gardan, J.P. Jung et A. Zakari. *Deux approches pour la représentation sous contraintes, d'objets complexes et évolutifs.* MICAD '92, vol. 1. pp 145-164.

[GGJZ 92b] H. Gaoua, Y. Gardan, J.P. Jung et A. Zakari. *Resolutions and representations of constraints on geometric and evolutive objects.* Eighth International Conference on CAD / CAM Robotics and Factories of the future. Metz, Aug. 92. pp 27-45.

[GJ 94] Y. Gardan, J.P. Jung. *La gestion des contraintes en CAO : propositions pour la modélisation et le dialogue.* Journées 3IA '94, Infographie Interactive et Intelligence Artificielle, Limoges France, Avril 94. pp 39-47.

[Gluc 75] H. Gluck. *Almost all simply connected closed surfaces are rigid.* Geometric Topology, pp 225-239. Lectures notes in Mathematics, num. 438, Springer Verlag, Berlin, 1975.

[GM 79] M. Gondran et M. Minoux. *Graphes et algorithmes.* Editions Eyrolles 1979.

[Gosl 83] J. Gosling. *Algebraic constraints.* Ph.D Thesis, Carnegie-Mellon University, May 83.

[GV 87] G.H. Golub et C.F. Van Loan. *Matrix Computations.* John Hopkins University Press, 1987.

[GZS 88] D. Gossard, R. Zuffante et H. Sukurai. *Representing dimensions, tolerances and features in MCAE systems.* IEEE Computer Graphics and Applications, Mars 88. pp 51-59.

[Hanr 83] P. Hanrahan. *Ray tracing algebraic surfaces.* Computer Graphics, vol.17, num. 3, July 83. pp 83-90.

[HB 78] R. Hillyard et I. Braid. *Characterizing non ideal shapes in terms of dimensions and tolerances.* Computer Graphics, vol. 12, num. 3, Aug. 78. pp 234-238.

[Hend 90] B. Hendrickson. *The molecule problem : determining conformation from pairwise distances.* Ph.D Thesis, Cornell University, 1990.

[Hend 92] B. Hendrickson. *Conditions for unique graphs realisations.* SIAM Journ. Computing, vol. 21, num. 1, Feb. 92. pp 65-84.

[HK 73] J.E. Hopcroft, R.M. Karp. *An $n^{5/2}$ algorithm for maximum matching in bipartite graphs.* SIAM J. Computing, vol. 2, num. 4, Dec. 73. pp 225-231.

[HT 73] J.E. Hopcoft et R.E. Tarjan. *Dividing a graph into triconnected components.* SIAM Journal on Computing, vol. 2, num. 3, Sep 73. pp 135-158.

[Inga 78] D.H. Ingalls. *The Smalltal-76 programming system : design and implementation.* Conference record of the fifth annual ACM symposium on principles of programming languages, Tucson Arizona, Jan. 78. pp 9-16.

[IRT 78] A. Itai, M. Rodeh et S.L. Tanimoto. *Some matching problems for bipartite graphs.* Journal of the Association of Computing Machinery, vol. 25, num. 4, Oct. 78. pp 517-525.

[Kear 87] R. B. Kearfott. *Some tests of generalized bisection.* ACM Transactions on Mathematical Software, vol. 13, num. 3, September 1987, 197-220.

[Knut 79] D.E. Knuth. *TEX and METAFONT : New directions in typesetting.* Digital Press, 1979.

[Kond 90] K. Kondo. *PIGMOD: parametric and interactive geometric modeller for mechanical design.* CAD vol.22, num.10, Dec 90. pp 633-644.

[Kond 92] K. Kondo. *Algebraic method for manipulation of dimensional relationships in geometric models.* CAD vol.24, num.3, March 92. pp 141-147.

[Kraw 69] R. Krawczyk. *Newton algorithmen zur bestimmung von nullstellen mit fehlerschranken.* Computing, 4, 1969. pp 187-201.

[Kutz 87] B. Kutzler. *Introduction to Bucheberger's Gröbner bases method.* Workshop on Computer-Aided Geometric Reasoning, INRIA, Sophia Antipolis, June 87. pp 7-25.

[LA 85] K. Lee et G. Andrews. *Inference of the positions of components in an assembly : part 2.* CAD vol.17, num.1, Jan. 85. pp 20-24.

[Lama 70] G. Laman. *On graphs and rigidity of plane skeletal structures.* Journal of Engineering Mathematics, vol.4, num.4, Oct. 70. pp 331-340.

[Laza 92] D. Lazard. *Systems of algebraic equations : algorithms and complexity.* Rapport interne LITP 92.20, Mars 92.

[LG 82] R. Light et D. Gossard. *Modification of geometric models through variational geometry.* CAD vol.14, num.4, July 82. pp 209-214.

[LGL 81] V. Lin, D. Gossard et R. Light. *Variational geometry in CAD.* Computer Graphics, vol.15, num.3, Aug.81. pp 171-178.

[LMMP 89] M. Lucas, D. Martin, P. Martin et D. Plemenos. *Le projet ExploFormes, quelques pas vers la modélisation déclarative de formes.* Actes de GroPlan 1989.

[LP 86] L. Lovasz et M.D. Plummer. *Matching Theory.* North-Holland, 1986.

[LY 82] L. Lovasz et Y. Yemini. *On generic rigidity in the plane.* SIAM Journal on Algebraic and Discrete Methods, vol. 3, num. 1, Mars 82. pp 91-98.

[MJ 79] R.E. Moore et S.T. Jones. *Safe starting regions for iterative methods.* SIAM J. Numerical Analysis, vol. 14, num. 6, Dec. 77. pp 1051-1065.

[Moor 66] R.E. Moore. *Interval Analysis.* Prentice Hall, Englewood Cliffs, N.J., 1986.

[MQ 82] R.E. Moore, L. QI. *A Successive interval for nonlinear systems.* SIAM J. Numerical Analysis, vol. 19, num. 4, August 1982, 845-850.

[Muro 87] K. Murota. *Systems Analysis by Graphs and Matroids, Structural Solvability and Controllability.* Springer Verlag, 1987.

[Nels 85] G. Nelson. *Juno, a constraint-based graphics system.* Computer Graphics, vol. 19, num. 3, 1985. pp 235-243.

[OR 70] J.M. Ortega et W.C. Rheinboldt. *Iterative solution of nonlinear equations in several variables.* Academic Press, New York, 1970.

[Owen 91] J.C. Owen. *Algebraic solution for geometry from dimensional constraints.* Proceeding Symposium on Solid Modeling Foundations and CAD/CAM Applications, Austin Texas. June 91. pp. 397-407.

[PF 90] A. Poth, C.J. Fan. *Computing the block triangular form of a sparse matrix.* ACM Transactions on Mathematical Software, vol. 16, num. 4, December 1990, 303-324.

[PS 86] P. Prusinkiewicz et D. Streibel. *Constraint-based modeling of three-dimensional shapes.* Graphics Interface '86, Vision Interface '86, 1986. pp 158-163.

[PS 93] A. Perez et D. Serrano. *Constraint base analysis tools for design.* Second Symposium on Solid Modeling Foundations and CAD/CAM Applications, Montreal Canada. Mai 93. pp. 281-290.

[Recs 86] A. Recski. *Matroids Theory and its Applications.* Springer Verlag, Berlin, 1986.

[RL 87] D.N. Rocheleau et K. Lee. *System for interactive assembly modelling.* CAD vol. 19, num.2, Mars 87. pp 65-72.

[Roll 91] D. Roller. *An approach to computer parametric design.* CAD, vol. 23, num. 5, June 91. pp 385-391.

[SAK 90] H. Suzuki, H. Ando et F.Kimura. *Geometric constraints and reasoning for geometrical CAD systems.* Computer and Graphics, vol. 14, num. 2, Oct. 90. pp 211-224.

[SAK 93] N. Sridhar, R. Agrawal et G.L. Kinzel. *Active occurence-matrix-based approach to design decomposition.* CAD vol. 25, num.8, Aug. 93. pp 500-512.

[Saxe 79] J.B. Saxe. *Embeddability of weighted graphs in k-space is strongly NP-hard.* Technical report, Computer Science Department, Carnegie-Mellon University, 1979.

[Schr 91] P. Schreck. *Modélisation d'une figure géométrique adaptée aux problèmes de constructions.* Actes de GroPlan 1991.

[Sene 90] P. Senechaud. *Bases de Gröbner booléennes : méthodes de calcul ; applications et parallélisation.* Thèse de doctorat, Institut National Polytechnique de Grenoble, 1990.

[Serr 91] D. Serrano. *Automatic dimensioning in design for manufacturing.* Proceeding Symposium on Solid Modeling Foundations and CAD/CAM Applications, Austin Texas. June 91. pp. 379-386.

[Smec 90] Smeci : *manuel de référence.* ILOG, version 1.5, Nov. 1990.

[Snyd 92] J.M. Snyder. *Interval analysis for computer graphics.* Computer Graphics, vol. 26, num. 2, July 1992. pp 121-130.

[Sugi 80] K. Sugihara. *On redundant bracing in plane skeletal structures.* Bulletin of the Electronical Laboratory, num.44, 1980. pp 376-386.

[Sund 87] G. Sunde. *A CAD system with declarative specification of shape.* Eurographics workshop on Intelligent CAD systems, April 87. Noordwijkerhout, The Nederlands. pp 90-105.

[Suth 63] I.E. Sutherland. *SKETCHPAD: a man machine graphical communication system.* Ph.D Thesis, MIT, Mass. 1963.

[Tarj 72] R.E. Tarjan. *Depth first search and linear graph algorithms.* SIAM J. Computing, vol. 1, num. 2, 1972. pp 146-160.

[Verr 90] A. Verroust. *Etude de problèmes liés à la définition, la visualisation et l'animation d'objets complexes en informatique graphique.* Thèse d'Etat, Université de Paris-Sud, Centre d'Orsay, Décembre 1990.

[VSR 92] A. Verroust, F. Schonek et D. Roller. *Rule oriented method for paremetrized computer-aided design.* CAD, vol. 24, num. 3, Oct 92. pp 531-540.

[WBL 85] F. Winkler, B. Bucheberger et F. Lichtenberger. *Algorithm 628 : an algorithm for constructing canonical bases of polynomial ideals.* ACM transactions on Mathematical Software, vol. 11, num. 1, Mars 85. pp 66-78.

[Wink 88] F. Winkler. *Algorithms in polynomial ideal theory and geometry.* Second Austrian-Hungarian Informatics Conference, Retzhof Austria, Sep. 88.

[Wood 90] R.F. Woodbury. *Variations in solids : a declarative treatment.* Computer and Graphics, vol. 14, num. 2, Oct. 90. pp 173-188.

[Yemi 79] Y. Yemini. *Some theoretical aspects of position-location problems.* IEEE Symposium Foundation of Computer Science, San Juan, Puerto Rico, Oct. 79. pp 1-8.

MODELISATION GEOMETRIQUE PAR CONTRAINTES : QUELQUES METHODES DE RESOLUTION

Résumé :

Diverses techniques de modélisation sont utilisées en synthèse d'images et en CAO (conception assistée par ordinateur) pour produire des images réalistes et analyser les propriétés géométriques des objets solides modélisés. Cependant, malgré les progrès récents, la conception de formes géométriques reste une tâche complexe. Les objet géométriques que veut modéliser l'utilisateur doivent vérifier certaines propriétés, traditionnellement appelées contraintes. Les contraintes dans les modeleurs classiques n'avaient pas de représentation informatique et c'était à l'utilisateur de les gérer "manuellement" et d'assurer la "cohérence" en cas de modification. Pour pallier ces inconvénients, certains systèmes de modélisation fournissent des outils de spécification des formes par des contraintes géométriques (modélisation implicite ou non-impérative).

Nous proposons dans cette thèse deux méthodes de résolution du système de contraintes. La première méthode étudie les graphes bipartis sous-jacents aux systèmes d'équations correspondant aux contraintes. Nous montrons qu'il est possible de décomposer polynomialement ces systèmes en sous-systèmes sur-contraints (plus d'équations que d'inconnues), sous-contraints (plus d'inconnues que d'équations) et bien-contraints (autant d'équations que d'inconnues) à partir du graphe biparti. La deuxième méthode proposée étudie les différentes configurations induites par des contraintes de distances, d'angles et de tangences entre points, droites et cercles. Les entités géométriques sont déterminées par un algorithme de réduction de graphes et un système à base de règles.

Mots-clés :

Modélisation géométrique, contraintes géométriques, graphes bipartis, couplage, graphe rigide, graphe des contraintes, réduction de graphes.