Rapport de stage de D.E.A.

Méthodes matricielles pour la résolution d'équations polynômiales et application au problème inverse 6R.

Olivier Ruatta
18 juin 1999

Dirigé par :
Bernard Mourrain

D.E.A. M.D.F.I.

I.N.R.I.A. Sophia-Antipolis
SAGA

Table des matières

1 Introduction **3**

2 Méthodes de résolution de systèmes d'équations polynômiales **7**
2.1 Introduction 7
2.2 Liens géométrie-algèbre 8
2.3 Aperçu sur les bases de Groebner 8
2.4 Aperçu sur les résultants 9
2.4.1 Généralités 9
2.4.2 Notion de résultant 10
2.4.3 Un exemple simple 10
2.4.4 Matrices de résultant 11
2.5 Méthodes purement matricielles 13
2.5.1 Introduction 13
2.5.2 Détermination des racines d'un polynôme d'une variable 13
2.6 Cas de plusieurs variables 15
2.6.1 Idéaux de dimension zéro 15
2.6.2 Passage au dual 15
2.6.3 Opérateur de multiplication 16
2.6.4 Calcul des racines par les vecteurs propres 16
2.7 Quelques méthodes d'analyse matricielle 17
2.7.1 Introduction 17
2.8 Décomposition en valeurs singulières 17
2.8.1 Problème de valeurs et vecteurs propres généralisé . . 18

3 Description du problème inverse 6R **20**
3.1 Introduction 20
3.1.1 Matrice de transformation homogène 21
3.1.2 Paramétrage de Denavit-Hartenberg 22
3.1.3 Le problème géométrique inverse 6R 24
3.2 Une première approche 25

4 Calcul effectif des solutions **32**
4.1 Partie calcul symbolique 32
4.2 Partie numérique 33
4.2.1 Calcul de la matrice Σ 33

Bibliographie **36**
.1 Preuves des lemmes 1, 2, 3 et 4 37
.2 Le code MAPLE 41
.3 Le code ALP 46
.3.1 Fichier principal 46
.3.2 Fichier de déclaration des types et d'affectation des matrices 49
.3.3 Déclarations de fonctions spécifiques : mymat.H 50
.3.4 Le fichier mymat.C 52

Chapitre 1

Introduction

La résolution de système d'équations polynômiales trouve beaucoup d'applications : la robotique, la vision, la chimie moléculaire par exemple. Au cours de ces dernières décennies, ces méthodes ont beaucoup évolué. La méthode la plus couramment utilisée par l'ingénieurie est la méthode de Newton. Cette dernière pose deux problèmes : elle ne converge pas nécessairement et ne donne pas toutes les solutions systématiquement. Les progrès les plus importants sont dus aux recherches en calcul formel. Deux grands types de méthodes se dégagent : celles utilisant les bases de Groebner pour l'explicitation des solutions, et les méthodes matricielles. Il existe d'autres méthodes que nous n'aborderons pas ici. Le succès des méthodes matricielles s'est renforcé dernièrement car elles permettent une évaluation rapide de solutions approchées. De plus ces méthodes présentent l'avantage d'être numériquement stables (contrairement aux méthodes purement formelles). Elles permettent donc de mener des calculs même s'il existe une incertitude sur les données. C'est en cela qu'elles sont bien adaptées aux problèmes auxquels nous nous sommes intéressé. Nous avons commencé par étudier un problème classique de la robotique qui a été largement abordé auparavant : *le problème inverse 6R.*

Dans ce problème, on considère un bras manipulateur série, c'est-à-dire une chaîne de solides reliés entre eux par des liaisons pivots (ce sont des liaisons qui ont un seul degré de liberté en rotation autour d'un axe). On considère une septième liaison, qui est celle du manipulateur (ou main). On connaît un certain nombre de paramètres comme la distance entre deux liaisons consécutives et surtout la position de la main par rapport à la première liaison. Nous désirons savoir combien de configurations du bras aboutissent à cette position relative entre la main et la première liaison. On sait depuis [5] que dans le cas le plus général, ce problème a au plus 16 solutions (sous-entendues réelles). Pour expliciter ces 16 solutions, les méthodes les plus utilisées consistent à déterminer un polynôme de degré 16 dont un des paramètres qui nous intéresse est racine. Ainsi, connaissant ce polynôme don-

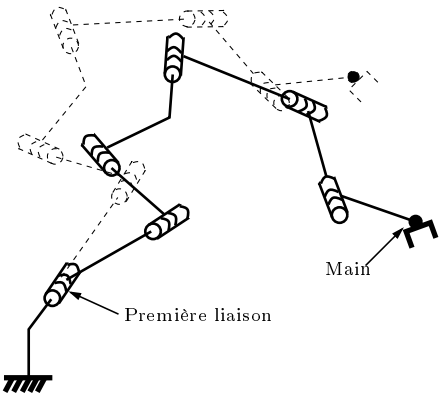

FIG. 1.1 – Un robot série à six liaisons permettant chacune une rotation

nant 16 valeurs possibles pour ce paramètre, nous avons la certitude de les avoir toutes décrites. La façon de trouver ce polynôme fait appel à différentes techniques d'élimination de variables, qui permettent par substitution et en remontant les calculs de déterminer au fur et à mesure les autres inconnues.

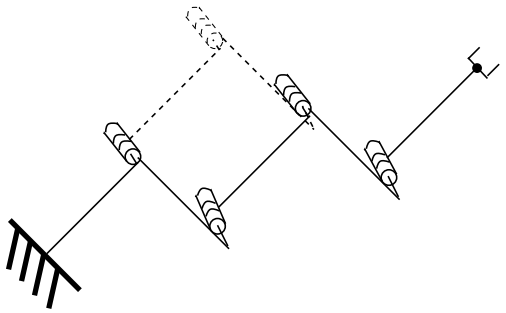

FIG. 1.2 – Deux configurations différentes pour une même position relative.

Nous avons repris une partie de ce raisonnement et utilisé d'autres techniques de résolution et d'élimination pour faire un programme simple, rapide et modulaire de résolution informatique de ce problème. Il est basé sur un algorithme symbolique et numérique. Grâce au calcul symbolique (effectué en utlisant MAPLE) nous nous ramenons à décrire une équation matricielle. Les matrices de l'équation prennent en compte tous les paramètres (connus ou inconnus). On génère automatiquement les matrices mises sous forme de code C++. La partie symbolique n'est donc exécutée qu'une fois. A ce stade nous sommes ramené à un problème de manipulation de matrices et de résolutions numériques pour lequel nous utilisons la bibliothèque ALP (spé-

cifique aux structures de données pour l'algèbre linéaire et polynômiale). La modularité (c'est-à-dire la possibilité de réutiliser et d'adapter le code) vient du fait que ALP est une bibliothèque C++, langage qui est conçu à ces fins. Ce besoin de modularité vient du fait que le problème *inverse 6R* n'est pas notre objectif principal. Notre but est d'adapter cette méthode à un autre problème quasiment analogue issu de la chimie.

Nous considérons une molécule comme un ensemble de liaisons et d'atomes. Les chimistes sont souvent confrontés à des problèmes de positions d'atomes. Par exemple, on s'intéresse au cas d'une cellule (biologique) dont on sait qu'elle fixe une molécule donnée grâce à un récepteur par l'intermédiaire d'atomes de la molécule bien positionnés par rapport à des sites donnés du recepteur. Pour certaines pathologies, on bloque ce phénomène en introduisant d'autres molécules contenant des atomes se fixant plus facilement que les précédents aux sites. Il faut alors que les nouvelles molécules portent ces atomes dans une configuration adéquate (voir figure 1.3).

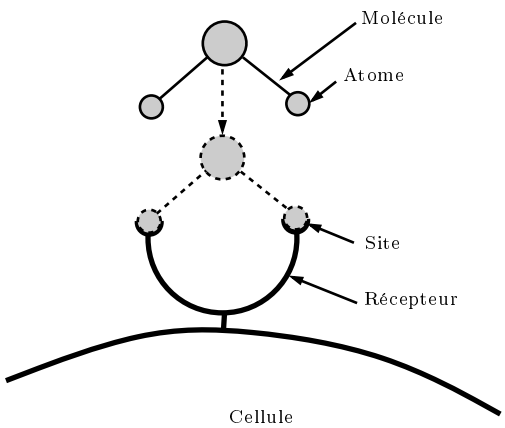

FIG. 1.3 – La molécule se fixe à la cellule grâce au récepteur .

On peut aussi vouloir remplacer une séquence d'atomes indésirables dans une molécule par une séquence neutre choisie sans changer fondamentalement la structure de la molécule. Mais là encore des impératifs géométriques entrent en jeu (voir figure 1.4).

Dans beaucoup de problèmes de la chimie moderne, des problèmes font intervenir de telles contraintes. Les chimistes ont déjà un savoir-faire en la matière, mais leurs méthodes ne sont ni automatisables, ni assez précises. Il se font une idée de la pertinence d'avoir une molécule cherchée, puis ils essaient de la synthétiser. Or la synthèse de molécules est un procédé long et coûteux. Les méthodes que nous avons développées s'appuient sur un formalisme qui semble être adaptable aux problèmes de géométries des molécules. Ainsi, à l'aide du programme réalisé, nous proposons un outil éventuellement plus

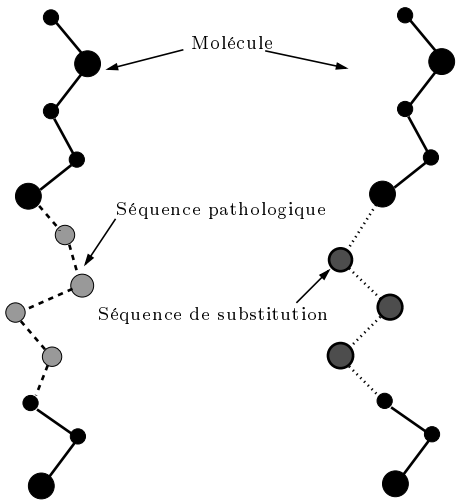

FIG. 1.4 – On veut substituer une séquence.

précis et plus rapide pour décrire les configurations prises par une molécule.

Nous devrions traiter à l'aide de ce programme une base de données de molécules de la société SANOFI afin de trier ces molécules.

Dans ce rapport nous décrirons donc les différentes méthodes mathématiques mises en jeu. Le choix des méthodes matricielles est motivé par le fait que les données de chimie sont souvent connues avec une incertitude. Puis nous exposerons le formalisme qui nous permet d'aborder le problème inverse 6R, avant d'exposer l'algorithme proposé. Enfin nous expliquerons comment adapter le formalisme précédemment introduit aux problèmes moléculaires.

Chapitre 2

Méthodes de résolution de systèmes d'équations polynômiales

Quelques notations et conventions

Dans toute la suite k est un corps commutatif, éventuellement algèbriquement clos. On note par $k[x_1, \ldots, x_n]$ l'algèbre des polynômes à n indéterminées sur k. Si $P \in k[x_1, \ldots, x_n]$ est tel que $P = \sum_{(i_1,\ldots,i_n)\in S} a_{i_1,\ldots,i_n} x_1^{i_1} \ldots x_n^{i_n}$, on appelle multidegré de P l'entier $deg\,(P) = \max_{(i_1,\ldots,i_n)\in S} \{i_1 + \ldots + i_n\}$.

2.1 Introduction

Pour le calcul des racines d'un système d'équations polynômiales, le premier objet introduit fut le résultant de deux polynômes. Puis sous l'impulsion de la géométrie algèbrique moderne, on s'est intéressé aux bases de Groebner, qui donne une représentation des idéaux d'une algèbre de polynômes. Ainsi on pouvait utiliser des méthodes de substitutions symboliques permettant un calcul exact. Plus récemment, on s'est intéressé de nouveau au résultant et surtout à ses généralisations, car ils se sont révélés accessibles par des méthodes d'algèbre linéaire supprimant ainsi un des principaux inconvénients des méthodes s'appuyant sur le calcul d'une base de Groebner (ce dernier calcul est en général très long). Nous expliquerons donc brièvement les deux méthodes, mais nous ne développerons que la deuxième car elle donne lieu à des méthodes purement matricielles dont certaines, semi-numériques, rapides. Mais avant expliquons un d'où viennent ces méthodes, c'est-à-dire les bases de la géométrie algébrique.

2.2 Liens géométrie-algèbre

Quand on considère un système d'équations polynômiales $p_1, p_2, \ldots, p_m \in k[x_1, \ldots, x_n]$, on peut se demander combien il a de solutions dans k^n, quelle est la géométrie de ces solutions... L'exemple le plus connu est le cercle unité dans R^2 qui est l'ensemble des solutions de l'équation $x^2 + y^2 = 1$. La géométrie algèbrique a pour objet l'étude de tels objets. Cette branche des mathématiques a pris un essor considérable ; elle s'est énormement ramifiée et enrichie au cours du siècle et est maintenant une des disciplines majeures des mathématiques, et ce d'autant plus qu'elle a de nos jours de très nombreuses applications. La géomètrie algébrique classique a fabriqué des outils permettant de lire les propriétés des solutions d'un système d'équations polynômiales $V(f_1, \ldots, f_s) = \{(x_1, \ldots, x_n) \in k^n | f_1(x_1, \ldots, x_n) = \ldots = f_s(x_1, \ldots, x_n) = 0\}$, pour $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$, grâce aux propriétés de l'idéal qu'engendrent ces polynômes $< f_1, \ldots, f_s > \subset k[x_1, \ldots, x_n]$. Pour un aperçu de la problèmatique, voir [11], [2] et [4]. De la tendance moderne des mathématiques et du besoin des applications, une théorie effective s'est développée. C'est-à-dire qu'on a cherché à calculer les solutions de systèmes d'équations polynômiales, et non seulement à regarder leurs propriétés géométriques ou algébriques. Toutes ces théories rendent les calculs effectifs dans l'algèbre quotient. C'est dans cette algèbre qu'on établit un dictionnaire entre les propriétés géomètriques et les propriétés algèbriques (théorie de la dimension, théorie de l'intersection, ...).

2.3 Aperçu sur les bases de Groebner

Considérons l'anneau $k[x_1, \ldots, x_n]$ et $I =< f_1, \ldots, f_p > \subset k[x_1, \ldots, x_n]$ un idéal de cet anneau. Les méthodes de bases de Groebner (aussi dites de bases standard) ont pour objet de décrire une base de $k[x_1, \ldots, x_n]/I$ en le considérant comme une k-espace vectoriel (par exemple $\{1, X, Y\}$ est une base de $k[X, Y]/ < X^2, Y^2, XY >$). Cette description nous permet de "linéariser" le problème. En effet, la plupart des propriétés de $k[x_1, \ldots, x_n]/I$ se lisent sur la base. Le résultat principal de cette théorie dit qu'une telle base existe toujours et qu'elle admet un nombre fini d'élément. Le problème c'est que pour avoir de bonnes propriétés vis-à-vis de la division, on est obligé de se fixer un ordre sur les monômes (appelé ordre monômial). A priori, pour deux ordres monômiaux distincts on a deux bases différentes. C'est hélas vrai en général. De plus les calculs peuvent être plus ou moins faciles en fonction de la base. Néanmoins cet outil de description du quotient permet de développer des algorithmes de calcul effectifs dans $k[x_1, \ldots, x_n]/I$. Ainsi, la description de bases de Groebner a pris une place centrale en théorie de l'élimination (dont l'objectif est de faire de la substitution de variables de façon analogue au pivot de Gauss en algèbre linéaire) et par là-même pour les calculs effectifs

en théorie de l'intersection (qui vise à décrire la variété algèbrique qui est l'intersection de plusieurs autres variétés). L'algorithme le plus répandu pour le calcul de telles bases est dû à Buchberger. Il se caractérise néanmoins par une forte complexité et les calculs sont souvent très longs ou trop lourds, et ne sont pas numériquement stable en cas d'incertitude sur les coefficients des polynômes (Pour plus d'information voir [2]).

2.4 Aperçu sur les résultants

2.4.1 Généralités

Le formalisme de la géométrie algèbrique trouve un cadre naturel avec la géométrie projective. Il y a plusieurs raisons à cela, principalement car dans un espace projectif il existe un théorème énonçant que la projection d'une variété algébrique qui y est plongée est aussi une variété algébrique. Or ce résultat est généralement infirmé dans le cadre affine. Par exemple dans R^2, on considère la variété $V = V(XY - 1 = 0)$. On considère alors la projection de cette variété sur l'axe $\{Y = 0\}$ (on note Π la projection). L'ensemble $\Pi(V)$ n'est pas une variété algébrique. Il est donc courant de *plonger* l'espace affine

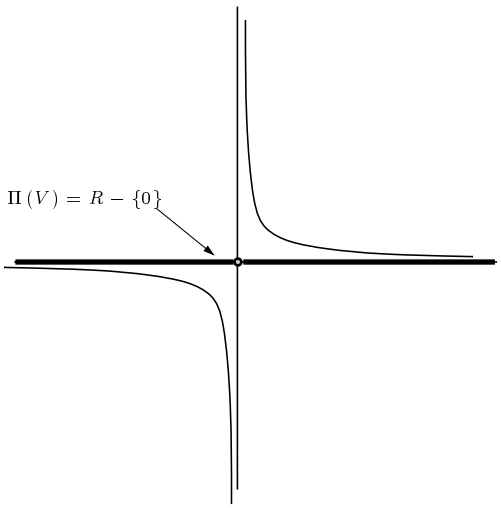

FIG. 2.1 – Projection de V

ambiant dans un espace projectif. On utilise les applications

$$\Phi : \left\{ \begin{array}{ccc} k^n & \longrightarrow & P^n(k) \\ (x_1, \ldots, x_n) & \longmapsto & [x_1 : \ldots : x_n : 1] \end{array} \right.$$

et

$$\Psi : \left\{ \begin{array}{ccc} P^n(k) & \longrightarrow & k^n \\ [x_1 : \ldots : x_n : t] & \longmapsto & (x_1/t, \ldots, x_n/t) \end{array} \right.$$

pour passer d'un cas à l'autre. Si on note $U = \{[x_1 : \ldots : x_n : t] \in P^n(k) \,|t = 1\}$ alors $\Psi|_U$ et Φ sont des bijections réciproques l'une de l'autre entre U et k^n. Ainsi on démontre des résultats dans $P^n(R)$, on regarde ce que donne ce résultat sur U et on "redescend" ces résultats sur k^n. Ce n'est bien sûr pas aussi simple que cela. En effet, dans $P^n(k)$ les variétés sont définies par des polynômes homogènes (ce sont des polynômes dont les monômes ont tous le même multidegré) pour des raisons de cohérence avec la notion d'espace projectif (dans un espace projectif, $[\lambda * x_1 : \ldots : \lambda * x_n : \lambda * t] = \lambda * [x_1 : \ldots : x_n : t] = [x_1 : \ldots : x_n : t]$). On définit alors une opération qui a un polynôme de $k[x_1, \ldots, x_n]$ associe un polynôme homogène de $k[x_1, \ldots, x_n, t]$ qu'on appelle homogénéisation. Par exemple l'homogénéisé de $P(X, Y) = XY - 1$ est $P_\#(X, Y, t) = XY - t^2$. On étudie donc la variété associé à l'homogénéïsé de dans l'espace projectif. Si on a plusieurs équations il se peut que des variétés qui ne se coupaient pas dans le cas affine (i.e. des polynômes premiers entre eux) se coupent dans le cas projectif.

2.4.2 Notion de résultant

Soit X une variété algèbrique. On considère un système de s polynômes : $f_1 = f_2 = \ldots = f_s = 0$. On note k_i le nombre de coefficients de f_i. Alors si on considère tous les coefficients de ces polynômes comme des variables indépendantes, chaque polynôme f_i définit une équation $f_i(x, c_i)$ sur $X \times P^{k_i}$, où c_i représente les coefficients dans P^{k_i}. Le système d'équations polynômiales de départ nous définit une variété W_X. On considère alors le diagramme :

$$\begin{array}{rl} W_X = & \left\{(x, c_1, \ldots, c_n) \in X \times P^{k_1} \times \ldots \times P^{k_s} | f_i(x, c_i) = 0, \forall i\right\} \to X \\ \downarrow & \\ Z \subset & P^{k_1} \times \ldots \times P^{k_s} \end{array}$$

Dans $P^{k_1} \times \ldots \times P^{k_s}$, la variété W_X se projette sur une variété Z dont l'équation est le résultant. Un théorème (le théorème des fibres linéaires) nous garantit qu'il s'agit bien d'une hypersurface (qui admet donc une équation) dans ce cadre. Or on s'est plongé dans des espaces projectifs si bien qu'on a pu introduire de l'information parasite. Il y a de nombreuses façons de ce plonger dans des espaces projectifs, et le but est de trouver des méthodes qui aide à obtenir une variété la plus petite possible (afin d'introduire un minimum de solutions indésirables).

2.4.3 Un exemple simple

Soient P de degré p et Q de degré q deux polynômes d'une seule variable. On note par $k[X]_i$ l'ensemble des polynômes de $k[X]$ de degré inférieur ou égal à i. L'ensemble $k[X]_i$ est un sous-espace vectoriel de $k[X]$, et est donc un k-espace vectoriel.

Définition 1 *On définit*

$$\Phi_{P,Q} : \left\{ \begin{array}{ccc} k[X]_q \times k[X]_p & \rightarrow & k[X]_{p+q} \\ (U,V) & \longmapsto & UP+VQ \end{array} \right.$$

Proposition 1 *L'application $\Phi_{P,Q}$ est une application linéaire.*

On considère $\{1, X, \ldots, X^q\}$ comme base pour $k[X]_q$ et $\{1, X, \ldots, X^p\}$ pour $k[X]_p$, donc $\{1, X, \ldots, X^q, 1, X, \ldots, X^p\}$ est une base de $k[X]_q \times k[X]_p$ et $\{1, \ldots, X^{n+m}\}$ en est une pour $k[X]_{p+q}$. Dans ces bases, la matrice de l'application linéaire est la matrice de Sylvester de ces deux polynômes et est donnée par :

$$S(P,Q) = \begin{pmatrix} a_0 & \ldots & 0 & b_0 & \ldots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \vdots & & a_0 & \vdots & & b_0 \\ \vdots & & \vdots & \vdots & & \vdots \\ a_p & & \vdots & b_q & & \vdots \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \ldots & a_p & 0 & \ldots & b_q \end{pmatrix}$$

Théorème 1 (Lemme de Bezout) *Les polynômes P et Q ont une racine commune si et seulement si* $\det(S(P,Q)) = 0$.

Les deux polynômes ont donc un facteur commun non trivial si et seulement si le déterminant de la matrice de cette application linéaire est nul. Or le déterminant de la matrice de Sylvester de ces deux polynômes est un polynôme en les coefficients de ces derniers. On obtient ainsi une contrainte polynômiale sur les cofficients pour que P et Q aient une racine commune. Cette équation est d'ailleurs connue sous le nom de résultant de ces deux polynômes.

2.4.4 Matrices de résultant

Principe de construction

Beaucoup de méthodes pour construire le résultant utilisent des matrices de résultant. Une telle matrice est telle que son déterminant soit un multiple non trivial du résultant. C'est une façon efficace de calculer le polynôme résultant et de résoudre des systèmes d'équation s polynômiales. Un exemple qui fut le premier connu est la matrice de Sylvester dans le cas de deux polynômes d'une variable, qui est décrit dans l'exemple du début de section. Les travaux de F.S. Macaulay permettent de généraliser la construction de Sylvester dans le cas de plusieurs variables. L'optique est de "linéariser" le problème. Pour un système $f_1 = \ldots = f_{n+1} = 0$, comme pour le cas d'une

variable, on veut exprimer les images des images des monômes par les f_i. Le support d'un polynôme est un ensemble des multi-indices contenant celui des coefficients non nuls du polynôme. On note par A_i un support de f_i. On s'intéresse à l'application linéaire suivante :

$$S : \begin{cases} < x^{B_1} > \times \ldots \times < x^{B_{n+1}} > & \longrightarrow & < x^B > \\ (g_1, \ldots, g_{n+1}) & \longmapsto & \sum_{i=1}^{n+1} g_i f_i \end{cases}$$

L'ensemble B_i est l'ensemble de tous les indices des monômes possibles de la forme $\prod_{j=1, j\neq i}^{n+1} x^{\alpha_j}$ telles que $\alpha_j \in A_j$ et B est le support de tous les polynômes de la forme $x^b f_i$ pour $b \in B_i$. La matrice de résultant est celle associée à cette application linéaire dans les bases monômiales respectives. Alors un théorème nous dit que cette matrice est un multiple du résultant. On remarque qu'à chaque fois qu'on réduit l'ensemble support des polynômes, on diminue la taille de la matrice.

Matrice de Macaulay non homogène

Cette construction concerne le résultant de $n+1$ polynômes de $k[x_1, \ldots, x_n]$ de degré respectif $d_1, \ldots, d_n$. Soient $\nu = \sum_{i=1}^{n+1} d^i - n$ et soit x^B l'ensemble des monômes en $x = (x_1, \ldots, x_n)$ de degré inférieur ou égal à ν. Soient $x_n^{d_{n+1}} x^{B_{n+1}}$ l'ensemble des tous les monômes de x^B qui sont divisible par $x_n^{d_{n+1}}$. Dans le reste des monômes $x^B - x_n^{d_{n+1}} x^{B_{n+1}}$, on note $x_{n-1}^{d_n} x^{B_n}$ ceux qui sont divisibles par $x_{n-1}^{d_n}$.On définit ainsi par induction une suite d'ensemble de monômes : $x_{i-1}^{d_i} x^{B_i}$. L'ensemble $x^B - x_n^{d_{n+1}} x^{B_{n+1}} - \ldots - x_1^{d_2} x^{B_2}$ est noté x_1^B et est défini par $x^{B_1} = \left\{ x_1^{b_1} x_2^{b_2} \ldots x_n^{b_n} | 0 \leq b_i \leq d_{i+1} - 1 \right\}$. Il contient $d_2 \ldots d_{n+1}$ monômes. On considère alors la matrice S associée aux sous-ensembles B_i, qui est carré de taille inférieure ou égale au nombre de monômes de degré $\leq \nu$ (i.e. $\begin{pmatrix} \nu + n \\ n \end{pmatrix}$). Alors la matrice S considérée et celle de l'application linéaire décrite dans la partie précédente exprimée dans les bases associées aux supports B_i donnés construits ici. F.S. Macaulay a montré que le résultant du système est le quotient du déterminant de S par le déterminant d'une sous-matrice de S.

2.5 Méthodes purement matricielles

2.5.1 Introduction

Ces méthodes permettent l'étude des système d'équations polynômiales au travers d'opérateurs linéaires dans l'algèbre quotient. Soit $f_1 = f_2 = \ldots = f_m = 0$ un système d'équations polynomiales. On note $I =< f_1, f_2, \ldots, f_m >$ l'idéal engendré par ces polynômes, on donnera une description de $k[x_1, \ldots, x_n]/(I)$ grâce à l'opérateur associé à la multiplication par un polynôme.

2.5.2 Détermination des racines d'un polynôme d'une variable

On considère un polynôme $P \in k[x]$, avec $P(x) = \sum_{i=0}^{i=d} a_i x^i$ de degré d ($a_d \neq 0$). On supposera k algébriquement clos. On note $A = k[x]/(P)$, c'est un espace vectoriel de dimension d et de base $\left(1, x, \ldots, x^{d-1}\right)$. Supposons dans un premier temps que toutes les racines $\zeta_i \in k$ de P soient simples : $P(x) = a_d \prod_{i=0}^{d-1} (x - \zeta_i)$, avec $\zeta_i \neq \zeta_j$ si $i \neq j$.

Considérons alors l'opérateur de multiplication par x dans A :

$$M_x = \begin{Bmatrix} 1 \\ x \\ x^2 \\ \ldots \\ x^{d-1} \end{Bmatrix} \begin{pmatrix} \{ & 1 & x & \ldots & x^{d-1} & \} \\ & 0 & 0 & \ldots & -\frac{a_0}{a_d} \\ & 1 & 0 & \ldots & -\frac{a_1}{a_d} \\ & 0 & 1 & \ldots & -\frac{a_2}{a_d} \\ & \vdots & \ddots & \ddots & \vdots \\ & 0 & 0 & \ldots & -\frac{a_{d-1}}{a_d} \end{pmatrix}$$

Cette matrice est connue dans la littérature sous le nom de matrice compagnon, dont la dernière colonne correspond au reste de la division euclidienne de x^d par P. Le polynôme caractéristique de M_x est P et ses valeurs propres sont les racines ζ_i de P. Comme ces valeurs propres sont distincts, la matrices est diagonalisable. Ce qui implique que pour tout $Q \in k[x]$ la matrice de la multiplication par Q sur A a pour valeurs propres les évaluations de Q aux racines de P : les valeurs propres de M_Q sont les $Q(\zeta_i)$, pour $i = 0, \ldots, d-1$ car $M_Q = Q(M_x)$. Donc les matrices M_Q sont diagonalisables dans une même base, que nous allons maintenant expliciter. On note

$$e_i(x) = \prod_{\substack{j=0 \\ i \neq j}}^{d-1} \frac{x - \zeta_j}{\zeta_i - \zeta_j}$$

le i-ième polynôme d'interpolation de Lagrange associé aux racines de P. Pour tout i et$j \neq i$ on vérifie que $e_i(x)\, e_j(x)$, $e_i(x)\,(e_i(x)-1)$ et $(1-\zeta_i)\, e_i(x)$ s'annulent aux racines ζ_i de P. Ces polynômes sont donc divisibles par P, ce qui implique :

$$e_i e_j \equiv 0,\ e_i^2 = e_i,\ xe_i = \zeta_i e_i$$

dans A. De plus nous avons $e_0 + e_2 + \ldots + e_{d-1} = 1$. On en déduit que les $(e_i)_{i=0,\ldots,d-1}$ forment une base d'idempotents orthogonaux :

$$A = ke_1 \oplus ke_2 \oplus \ldots \oplus ke_{d-1}$$

La structure de l'agèbre quotient, dans le cas d'une variable où les racines du polynômes sont simples, est entièrement caractérisée en terme de ses idempotents, qui ne sont autres que les racines du polynômes par lequel on quotiente.

Soit $\hat{A} = L(A,k)$ l'algèbre dual de A. C'est un k-espace vectoriel de dimension d. On note $\left(d^i\right)_{i=0,\ldots,d-1}$ la base duale de $\left(x^i\right)_{i=0,\ldots,d-1}$. Alors, $\forall \Lambda \in \hat{A}$, on a :

$$\Lambda = \sum_{i=0}^{d-1} \Lambda\left(x^i\right) d^i$$

Quel que soit $g \in k[x]$, pour calculer $\Lambda(g)$ on calcule le représentant canonique de sa classe dans A : $g \equiv r = \sum_{i=0}^{d-1} r_i x^i$, alors dans $\hat{A}$, on a :

$$\Lambda(g) = \sum_{i=0}^{d-1} r_i \Lambda\left(x^i\right)$$

La matrice de

$$M_x^t : \left\{ \begin{array}{ccc} \hat{A} & \longrightarrow & \hat{A} \\ \Lambda & \longmapsto & \Lambda \circ M_x^t \end{array} \right.$$

est clairement la transposée de M_x. On note $1_\zeta \in \hat{A}$ l'application suivante : $\forall p \in A, 1_\zeta(p) = p(\zeta)$ où ζ est une racine de f. La formule d'interpolation de Lagrange nous dit que tout polynôme r de degré inférieur ou égale à $d-1$ s'écrit :

$$r(x) = \sum_{i=1}^{d} r(\zeta_i)\, e_i\left(x^i\right) = \sum_{i=1}^{d} 1_{\zeta_i}(r)\, e_i\left(x^i\right)$$

Donc les coefficients de r dans la base $(e_i)_{i=1..d}$ sont les $1_{\zeta_i}(r)$, donc $(1_{\zeta_i})_{i=1..d}$ est la base duale de $(e_i)_{i=1..d}$. Comme on a :

$$\forall p \in k[x],\ M_x^t\left(1_{\zeta_i}\right)(p) = 1_{\zeta_i}(xp) = \zeta_i p(\zeta_i)$$

Alors $M_x^t\left(1_{\zeta_i}\right) = \zeta_i 1_{\zeta_i}$. Donc 1_{ζ_i} est un vecteur propre de M_x^t.

Ce sont ces méthodes qu'on généralise dans le cas de plusieurs variable. C'est l'objet de la section suivante.

2.6 Cas de plusieurs variables

2.6.1 Idéaux de dimension zéro

On note $R = k[x_1, \ldots, x_n]$ et $I =< f_1, \ldots, f_m >$. On notera aussi $V(I)$ la variété definie par I dans k^n, K la clôture algébrique de k, I_K sera l'idéal engendré $< f_1, \ldots, f_m > \in K[x_1, \ldots, x_n]$ et $V_K(I)$ sera la veriété correspondante dans K^n. On définit aussi $A = k[x_1, \ldots, x_n]/(I)$ et $A_K = K[x_1, \ldots, x_n]/(I_K)$.

Un idéal est de dimension zéro si la variété qui lui est associé $V_K(I)$ est de dimension zéro, ce qui signifie qu'elle est finie. On note $V_K(I) = \{\zeta_1, \ldots, \zeta_n\}$. Soit $(x^\alpha)_{\alpha \in E}$ une base monômiale de A_K, et soit D le nombre d'élément de cette base. L'ensemble E est donc l'ensemble des multidegrés inférieurs à chacun des multidegré des f_i. Par exemple dans $C[X, Y]$ on considère l'idéal engendré par le polynôme : $P = \sum_{i+j=4} X^i Y^j$ alors $E = \{(i, j) \in Z^2 | i + j \leq 3$ et une base monômiale de cet idéal est $\left\{X^i Y^j\right\}_{i+j \leq 3}$.

2.6.2 Passage au dual

La dualité a une grande importance pour l'approche matricielle. On supposera désormais que $k = K$. Soient $A = R/(I)$ que nous supposerons de dimension D sur k, $(x^\alpha)_{\alpha \in E}$ une base de cette algèbre quotient ($E \subset N^n$). Le k-espace vectoriel des formes linéaires sur R sera noté $\hat{R}$ et celui des formes linéaires sur A sera noté $\hat{A}$. La base duale de $(x^\alpha)_{\alpha \in E}$ dans $\hat{A}$ est $(\delta^\alpha)_{\alpha \in E}$ et la base duale de celle de R, $(x^\alpha)_{\alpha \in N^n}$ est $(d^\alpha)_{\alpha \in N^n}$ dans $\hat{R}$. Toute forme linéaire $\Lambda \in \hat{A}$ s'écrit donc

$$\Lambda = \sum_{\alpha \in E} \Lambda(x^\alpha)\, \delta^\alpha$$

L'ensemble $\hat{A}$ s'identifie avec l'ensemble des élément de $\hat{R}$ s'annulant sur A. Remarquons que $\hat{A}$ est naturellement muni d'une structure de A-module comme suit :

$$\forall p \in A,\ \forall \Lambda \in \hat{A}, a.\Lambda : \left\{ \begin{array}{ccc} A & \longrightarrow & k \\ q & \longmapsto & p.\Lambda(q) = \Lambda(pq) \end{array} \right.$$

Parmi les éléments de $\hat{A}$, nous nous intéressons particulièrement à ceux de la forme :

$$1_\zeta : \left\{ \begin{array}{ccc} A & \longrightarrow & k \\ p & \longmapsto & p(\zeta) \end{array} \right.$$

où $\zeta \in V(I)$ est appelé évaluation en ζ.Comme $\zeta \in V(I)$, $1_\zeta \in \hat{A}$. On peut donc l'écrire sous la forme :

$$1_\zeta = \sum_{\alpha \in E} \zeta^\alpha \delta^\alpha$$

2.6.3 Opérateur de multiplication

Soit $p \in A$, nous allons nous intéresser à l'opérateur de multiplication par p dans A :

$$M_p : \left\{ \begin{array}{ccc} A & \longrightarrow & A \\ q & \longmapsto & M_p(q) = pq \end{array} \right.$$

L'endomorphisme précédent induit l'application k-linéaire suivante :

$$M_p^t : \left\{ \begin{array}{ccc} \hat{A} & \longrightarrow & \hat{A} \\ \Lambda & \longmapsto & M_p^t(\Lambda) = \Lambda \circ M_p \end{array} \right.$$

La matrice de M_p^t dans la base $(\delta^\alpha)_{\alpha \in E}$ est la transposée de la matrice de M_p. Ces deux endomorphismes ont les mêmes valeurs propres. Les résolution des système d'équations polynômiales par des méthodes matricielles est basée sur le théorème suivant :

Théorème 2 *1. Pour tout $p \in A$, les valeurs propres de M_p sont $p(\zeta_1), \ldots, p(\zeta_D)$, en particulier, les valeurs propres de M_{x_i} sont les i-ièmes coordonnées des éléments de $V(I)$.*

2. Les vecteurs propres communs à tous les endomorphismes $\left(M_p^t\right)_{p \in A}$ sont (à un scalaire non nul près) les éléments $1_{\zeta_1}, \ldots, 1_{\zeta_D}$ pour les valeurs propres $p(\zeta_1), \ldots, p(\zeta_D)$.

2.6.4 Calcul des racines par les vecteurs propres

Supposons que la base de A contienne $1, x_1, \ldots, x_n$. Soit $p \in R$ et M_p la matrice de multiplication dans cette base.

1. Calculer les vecteurs propres de $\Lambda = [\Lambda_1, \Lambda_{x_1} \ldots, \Lambda_{x_n}, \ldots]$ de $M_{x_i}^t$ dans la base duals de cette base.
2. Pour tout vecteur propre Λ pour lequel $\Lambda_1 \neq 0$, calculer $\zeta = \left(\frac{\Lambda_{x_1}}{\Lambda_1}, \ldots, \frac{\Lambda_{x_n}}{\Lambda_1}\right)$.

les points ζ ainsi calculés contiennent les racines simples de I (les points de $V(I)$ de multiplicité 1).

Le polynôme $p \in R$ est dit séparant si l'application :

$$\left\{ \begin{array}{ccc} V(I) & \longrightarrow & k \\ \xi & \longmapsto & p(\xi) \end{array} \right.$$

est injective. Si les racines sont simples et que p sépare les racines, alors l'algorithme ci-dessus donne exactement les éléments de $V(I)$.

2.7 Quelques méthodes d'analyse matricielle

2.7.1 Introduction

Les méthodes d'analyse matricielle sont développées dans le but d'évaluer numériquement des valeurs, des vecteurs propres, des vecteurs du noyau, de diagonaliser, de trianguler des matrices de nombres réels. Nous donnerons juste un petit aperçu des techniques que nous avons utilisé pour aborder les problèmes qui nous intéressaient.

2.8 Décomposition en valeurs singulières

Cette technique est souvent réalisée de manière itérative pour calculer les vecteurs du noyaux d'une matrice ou pour calculer son rang. Elle repose sue le théorème suivant :

Théorème 3 *Soient $A \in M_{m,n}$ et $q = \min\{m, n\}$. Il existe une matrice $D = (d_{i,j})$ avec $d_{i,j} = 0$ si $i \neq j$, et $d_{1,1} \geq d_{2,2} \geq \ldots \geq d_{qq} \geq 0$, et deux matrices unitaires $V \in M_m$ et $W \in M_n$ tels que $A = VDW^t$. Si $A \in M_{m,n}(R)$, alors V et W sont des matrices orthogonales réelles.*

Preuve La sphère euclidienne U de C^n est compacte et la fonction $f(x) = \|Ax\|_2$ est une fonction continue à valeurs réelles. D'après le théorème de Weierstrass, il existe $w \in U$ tel que $\|Aw\|_2 = \max_{x \in U}\{\|Ax\|_2\}$. Si $\|Aw\|_2 = 0$ alors $A = 0$ et le théorème est vérifié. Si $\|Aw\|_2 \neq 0$, on note $d_1 = \|Aw\|_2$ et $v = w/d_1 \in C^m$. Il existe $m-1$ vecteurs orthogonaux $v_2, \ldots, v_m \in U$ tels que la matrice $V_1 = (v, v_2, \ldots, v_m)$ soit unitaire et $n-1$ vecteurs orthogonaux $w_2, \ldots, w_n$ tels que la matrice $(w, w_2, \ldots, w_n)$ soit unitaire. Alors

$$\widetilde{A_1} = V_1^t A W_1 = \begin{pmatrix} v^t \\ v_2^t \\ \vdots \\ v_m^t \end{pmatrix} (Aw\ Aw_2\ \ldots\ Aw_n)$$

$$= \begin{pmatrix} v^t \\ v_2^t \\ \vdots \\ v_m^t \end{pmatrix} (d_1 w\ Aw_2\ \ldots\ Aw_n)$$

$$= \begin{pmatrix} d_1 & v^t A w_2 & \ldots & v^t A w_n \\ 0 & & & \\ \vdots & & A_2 & \\ 0 & & & \end{pmatrix}, A_2 \in M_{m-1,n-1}$$

$$= \begin{pmatrix} d_1 & z^t \\ 0 & A_2 \end{pmatrix}, z = \begin{pmatrix} v^t A w_2 \\ \vdots \\ v^t A w_n \end{pmatrix} \in C^{n-1}$$

Considérons maintenant le vecteur unitaire $\zeta = \begin{pmatrix} d_1 \\ z \end{pmatrix} / \left(d_1^2 + z^t z\right)^{1/2} \in C^n$ et calculons

$$\|A\left(W_1 \zeta\right)\|^2 = \| \left(V_1^t A W_1\right) \zeta \|^2 = \|\widetilde{A_1} \zeta\|^2$$
$$= \left(\left(d_1^2 + z^t z\right)^2 + \|A_2 z\|^2\right) / \left(d_1^2 + z^t z\right)$$
$$\geq d_1^2 + z^t z$$

qui est strictement plus grand que d_1^2 si $z \neq 0$. C'est une contradiction avec le fait que d_1 soit le maximum de f. Donc $z = 0$ et $\widetilde{A_1} = V_1^t A W_1 = \begin{pmatrix} d_1 & 0 \\ A_2 & 0 \end{pmatrix}$. On recommence alors le raisonnement avec A_2 et on obtient finalement les matrices V et W. La matrice $D = (d_{i,j})$ est telle que $d_{i,i} = d_i$ pour $i = 1, \ldots, q$.Si $m \leq n$, $AA^t = \left(VDW^t\right)\left(WDV^t\right) = VDDV^t = DD$ et

$$DD = diag\left(d_1^2, \ldots, d_q^2\right)$$

donc les réels positifs d_i sont uniquement déterminés par A ; si $n \leq m$ on arrive à la même conclusion. Si A est une matrice réelle, on effectue tous les calculs dans R, et les matrices unitaires V et W sont en fait orthogonales. ♣

Détermination du rang et du noyau d'une matrice à l'aide d'une telle décomposition

L'intérêt d'une telle décomposition est que le rang est le nombre de $d_{i,i}$ non nul, ce qui est immédiatement calculable si on connaît V et W puisque $V^t A W = D$. De plus si on a calculé le rang par ce moyen, que ce rang est r, alors comme $AW = VD$, les $m - r$ premières colonnes de W sont des vecteurs de base du noyau.

2.8.1 Problème de valeurs et vecteurs propres généralisé

On connaît des techniques itératives efficaces pour le calcul des valeurs et vecteurs propres de matrices (factorisation QR ou QZ par exemples). Nous allons ici expliquer comment généraliser ces méthodes à un cadre plus large. Ici on veut résoudre un système linéaire du type

$$Ax = \lambda B x$$

on se ramène alors à

$$\left(A - \lambda B\right) x = 0$$

C'est-à-dire à un problème de calcul de noyau dépendant du paramètre λ qui n'est pas trop difficile. L'intérêt de ce calcul est qu'il permet de résoudre des systèmes du type :

$$Cx = \lambda Bx + \lambda^2 Ax$$

En effet, on introduit alors le vecteur $\begin{pmatrix} x \\ \lambda x \end{pmatrix}$ et les matrices $M_1 = \begin{pmatrix} 0 & I \\ C & -B \end{pmatrix}$ et $M_2 = \begin{pmatrix} I & 0 \\ 0 & A \end{pmatrix}$ et notre problème revient à résoudre $1M_1 \begin{pmatrix} x \\ \lambda x \end{pmatrix} = \lambda M_2 \begin{pmatrix} x \\ \lambda x \end{pmatrix}$. Nous rencontrerons un tel problème ultérieurement.

Chapitre 3

Description du problème inverse 6R

3.1 Introduction

Le problème inverse 6R est un problème géométrique issu de la modélisation en robotique. On regarde un robot manipulateur comme un ensemble de solides reliés par des liaisons. Le problème de géométrie standard de ces

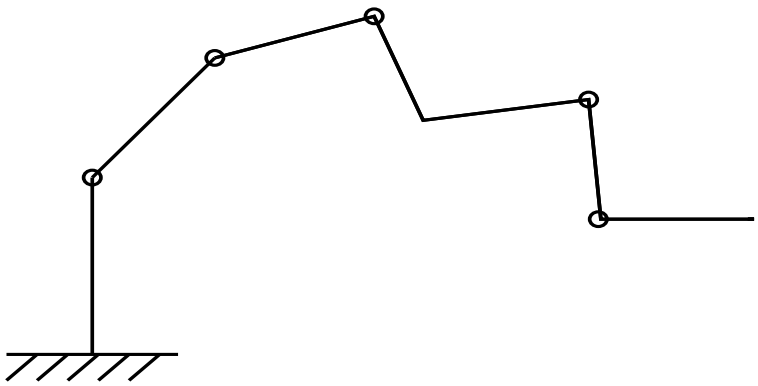

FIG. 3.1 – Bras manipulateur à 5 articulations (liaisons)

robots est de savoir quel est l'ensemble des positions accessibles au bout du bras connaissant les solides et les liaisons. Le problème inverse consiste à se donner la position du bout du bras, les solides et des paramètres sur les liaisons et à se demander s'il existe une configuration du robot donnant cette position au bout du bras.S'il existe une telle configuration, combien y en a-t-il ? Si cet ensemble de configurations n'est pas fini, comment le caractériser ?

3.1.1 Matrice de transformation homogène

Changements de coordonnées affines

On considère l'espace affine R^3, deux repères orthonormés direct $R = \left(O, \overrightarrow{i}, \overrightarrow{j}, \overrightarrow{k}\right)$ et $R' = \left(O', \overrightarrow{i'}, \overrightarrow{j'}, \overrightarrow{k'}\right)$ de cet espace. On note $R'' = \left(O, \overrightarrow{i''}, \overrightarrow{j''}, \overrightarrow{k''}\right)$, et A la matrice de passage de R à R''. Alors en notant $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ les coordonnées d'un point dans le repère R et $\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}$ les coordonnées de ce point dans R', on a :

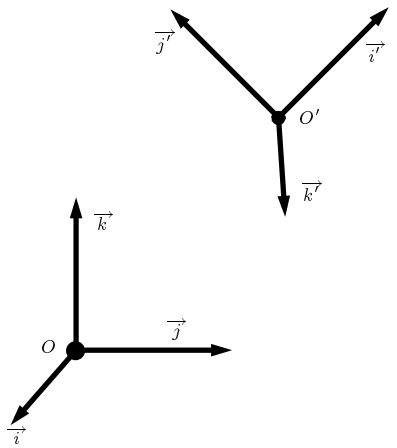

FIG. 3.2 – Changement de repères affines

$$A \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \overrightarrow{OO'} = \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} \tag{3.1}$$

Comme les repères sont orthonormés directs, A est une matrice orthogonale, i.e. $A^{-1} = A^t$.

Cas projectif, matrices de transformation homogène

On considère les applications $\Pi : \begin{cases} R^3 & \longrightarrow & P^3(R) \\ (x,y,z) & \longmapsto & [x:y:z:t] \end{cases}$, où $P^3(R)$ désigne l'espace projectif, et $\Phi : \begin{cases} P^3(R) & \longrightarrow & R^3 \\ [x:y:z:t] & \longmapsto & (x/t, y/t, z/t) \end{cases}$. On a clairement $\Phi \circ \Pi = Id_{R^3}$, autrement dit Π est une section de Φ. En notant $U = \{[x:y:z:t] \in P^3 | t = 1\}$, ces deux applications induisent une

bijection entre R^3 et U. L'ensemble U est naturellement muni d'une structure d'espace affine, et Π et $\Phi|_U$ sont des isomorphismes d'espaces affines. Dans toute la suite on identifiera R^3 et U. Dans U la relation 3.1 peut se réécrire plus simplement :

$$\begin{pmatrix} A & \overrightarrow{OO'} \\ 0\,0\,0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{3.2}$$

La relation 3.2 est un outil utile pour la description géométrique de bras manipulateurs. C'est l'objet du paragraphe suivant.

3.1.2 Paramétrage de Denavit-Hartenberg

Paramétrage pour la réalisation de modéles géométriques

On considère un ensemble de solides $S_1, \ldots, S_n$ reliés par des liaisons $L_{i,j}$ pour $1 < i < j < n$. On se place tout de suite dans le cadre qui nous intéresse, c'est-à-dire dans le cas d'un robot série (ou manipulateur série), où l'ensemble des liaisons est réduit à $L_{i,i+1}$ pour $i = 1, \ldots, n-1$.

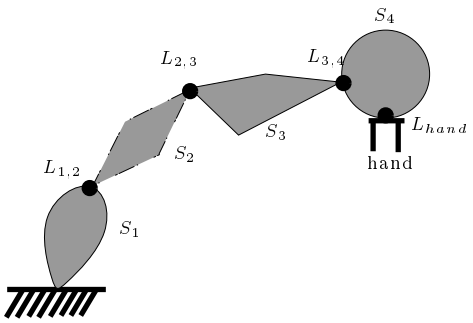

FIG. 3.3 – Représentation d'un bras manipulateur serie

Pour avoir une représentation géométrique, on associe à chaque liaison un repère affine "qui lui est attaché".

Ainsi connaissant les possibilités de chaque liaison (i.e. les positions possibles de cette liaison dans le repère "attaché" à la liaison précédente) il est possible de décrire les positions accessibles à la main au bout du bras.

Paramétrage de Denavit-Hartenberg

Le paramétrage de Denavit-Hartenberg consiste en un choix de repère particulier pour chaque liaison. On note R_l le repère associé à la liaison L_l, avec $R_l = \left(O_l, \overrightarrow{i_l}, \overrightarrow{j_l}, \overrightarrow{k_l}\right)$, avec les choix suivants :

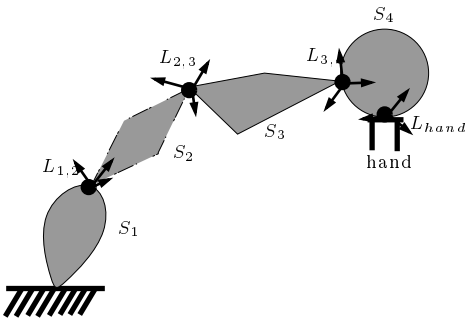

FIG. 3.4 – Bras manipulateur serie munie de repère

1. $\overrightarrow{k_l}$ est l'axe de la liaison entre S_{l-1} et S_l,
2. $\overrightarrow{i_l}$ est la perpendiculaire commune à $\overrightarrow{k_{l-1}}$ et $\overrightarrow{k_l}$,
3. $\overrightarrow{j_l} = \overrightarrow{k_l} \wedge \overrightarrow{i_l}$

Grâce à ces repères on définit deux groupes de paramètres :

1. 2 paramètres de localisation de $\overrightarrow{k_l}$ dans le solide S_{i-1} :
   (a) α_i qui est l'angle entre $\overrightarrow{k_{l+1}}$ et $\overrightarrow{k_l}$ dans le plan orthogonal à $\overrightarrow{i_{l-1}}$,
   (b) d_i qui est la longueur de la projection de $\overrightarrow{O_{l-1}O_l}$ sur la droite engendrée par $\overrightarrow{i_{l-1}}$ passant par O_{l-1},
2. deux paramètres de mouvement (rotation et/ou translation) de S_l par rapport à S_{l-1} (grâce à la liaison L_{l-1}) :
   (a) θ_l qui est l'angle entre $\overrightarrow{i_{l-1}}$ et $\overrightarrow{i_l}$ dans le plan orthogonal à $\overrightarrow{k_l}$,
   (b) r_l qui est la longueur de la projection de $\overrightarrow{O_{l-1}O_l}$ sur la droite engendrée par $\overrightarrow{k_l}$ passant par O_l,

Cette approche nous permet d'écrire la matrice de A_i de changement de repères qui fait passer de R_l à R_{l+1} de façon simple. En effet, pour ce faire, on commence par réaliser une rotation d'angle α_i autour de l'axe défini par $\overrightarrow{i_l}$, puis on réalise une translation de vecteur $r_l\,\overrightarrow{i_l}$, puis une rotation d'angle θ_i autour de l'axe défini par $\overrightarrow{k_l}$, et enfin on fait une translation de vecteur $d_i\,\overrightarrow{k_l}$. On remarque qu'on aurait pu commencer par la rotation et la translation

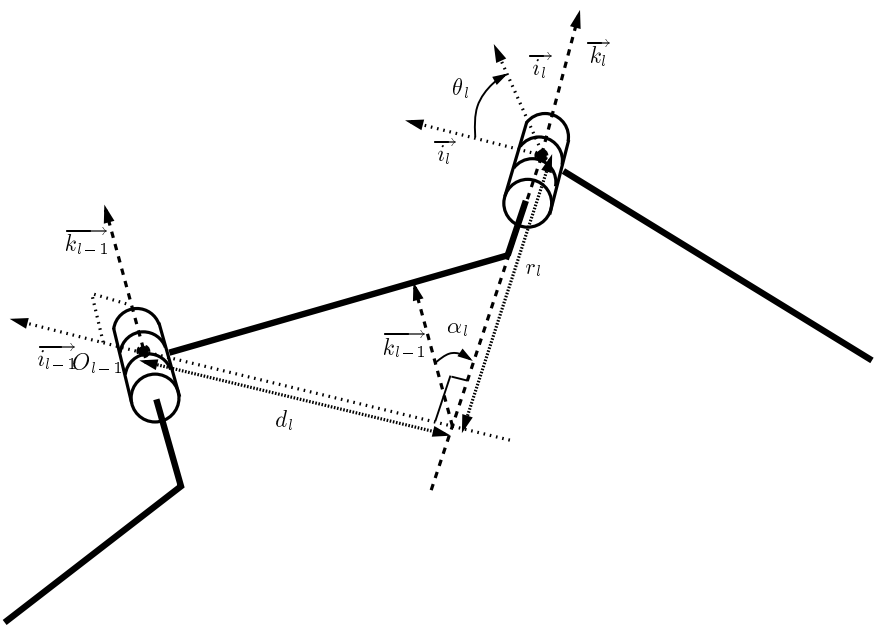

FIG. 3.5 – Visualisation des paramètres de denavit-Hartenberg

selon $\overrightarrow{k_l}$, puis faire la rotation et la translation selon $\overrightarrow{i_l}$. On a :

$$\begin{pmatrix} \cos\theta_l & -sin\theta_l & 0 & 0 \\ \sin\theta_l & \cos\theta_l & 0 & 0 \\ 0 & 0 & 1 & d_l \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & r_l \\ 0 & \cos\alpha_l & -\sin\alpha_l & 0 \\ 0 & \sin\alpha_l & \cos\alpha_l & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos\theta_l & -sin\theta_l\cos\alpha_l & \sin\theta_l\sin\alpha_l & r_l\cos\theta_l \\ \sin\theta_l & \cos\theta_l\cos\alpha_l & -\cos\theta_l\sin\alpha_l & r_i\sin\theta_l \\ 0 & \sin\alpha_l & \cos\alpha_l & d_l \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.3}$$

L'égalité 3.3 permet d'obtenir une expression simple pour la matrice à utiliser dans l'expression 3.2. Nous allons maintenant appliquer ce formalisme à un problème.

3.1.3 Le problème géométrique inverse 6R

Nous nous intéressons au problème d'un bras manipulateur ayant sept liaisons, qui sont toutes des liaisons pivots (chaque liaison introduit un degré de liberté en rotation autour de l'axe $\overrightarrow{k_l}$ lui correspondant). On suppose connus α_l, d_l, et r_l pour l allant de 1 à 7 et on se donne la matrice A_{hand} qui est la matrice de passage de R_7 à R_1. On voudrait savoir si, pour ces

données,il existe des valeurs pour les θ_l donnant une configuration possible pour le bras manipulateur. Si oui, il y en a peut-être plusieurs. Dans ce cas on veut savoir combien et lesquelles. Mais nous ne nous intéresserons qu'au cas où il y a un nombre fini de configurations qui sont solutions de ce problème. Si une telle solution existe alors elle est nécessairement solution du système :

$$A_1 A_2 A_3 A_4 A_5 A_6 = T \tag{3.4}$$

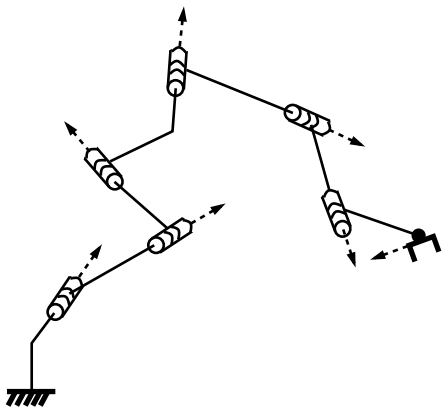

FIG. 3.6 – Le bras du problème inverse 6R

La relation 3.4 dit que si on réalise successivement les changements de repères de liaison en liaison jusqu'au bout du bras, on retrouve le changement de repères qui fait passer de la première articulation à la dernière (le bout du bras). On remarque que $r_i = 0$, pour tout i. Il existe plusieurs méthodes pour aborder ce problème. Nous commencerons par donner celle qui est donnée dans les articles [8] et [3], dont nous développerons le raisonnement et les méthodes dans la section suivante.

3.2 Une première approche

Fixons nous quelques notations : On note $Rot_i = \begin{pmatrix} \cos\theta_i & \sin\theta_i & 0 \\ \sin\theta_i & -\cos\theta_i & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $G_i = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -\cos\alpha_i & \sin\alpha_i \\ 0 & \sin\theta_i & \cos\theta_i \end{pmatrix}$, $\widetilde{a_i} = \begin{pmatrix} r_i \\ 0 \\ d_i \end{pmatrix}$, $M_{hand} = \begin{pmatrix} l_x & m_x & n_x \\ l_y & m_y & n_y \\ l_z & m_z & n_z \end{pmatrix}$ et $\widetilde{q} = \begin{pmatrix} q_x \\ q_y \\ q_z \end{pmatrix}$.

L'objectif de la méthode utilisée est de se ramener à un problème de système d'équations polynômiales bien contraint. Pour cela, nous commençons par réécrire la relation matricielle 3.4 de la façon suivante :

$$A_3 A_4 A_5 = A_2^{-1} A_1^{-1} T A_6^{-1} \tag{3.5}$$

En effet, toutes les matrices A_i sont inversibles (elles représentent des changements de bases) et l'expression de leur inverse est très simple. L'équation matricielle 3.5 peut se réécrire :

$$\phi(\theta_3, \theta_4, \theta_5) = \psi(\theta_1, \theta_2, \theta_6) \tag{3.6}$$

Ce qui peut s'exprimer facilement sous forme matricielle par :

$$\begin{pmatrix} \phi_{1,1}(\theta_3,\theta_4,\theta_5) & \phi_{1,2}(\theta_3,\theta_4,\theta_5) & \phi_{1,3}(\theta_3,\theta_4,\theta_5) & \phi_{1,4}(\theta_3,\theta_4,\theta_5) \\ \phi_{2,1}(\theta_3,\theta_4,\theta_5) & \phi_{2,2}(\theta_3,\theta_4,\theta_5) & \phi_{2,3}(\theta_3,\theta_4,\theta_5) & \phi_{2,4}(\theta_3,\theta_4,\theta_5) \\ \phi_{3,1}(\theta_3,\theta_4,\theta_5) & \phi_{3,2}(\theta_3,\theta_4,\theta_5) & \phi_{3,3}(\theta_3,\theta_4,\theta_5) & \phi_{3,4}(\theta_3,\theta_4,\theta_5) \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \psi_{1,1}(\theta_1,\theta_2,\theta_6) & \psi_{1,2}(\theta_1,\theta_2,\theta_6) & \psi_{1,3}(\theta_1,\theta_2) & \psi_{1,4}(\theta_1,\theta_2) \\ \psi_{2,1}(\theta_1,\theta_2,\theta_6) & \psi_{2,2}(\theta_1,\theta_2,\theta_6) & \psi_{2,3}(\theta_1,\theta_2) & \psi_{2,4}(\theta_1,\theta_2) \\ \psi_{3,1}(\theta_1,\theta_2,\theta_6) & \psi_{3,2}(\theta_1,\theta_2,\theta_6) & \psi_{3,3}(\theta_1,\theta_2) & \psi_{3,4}(\theta_1,\theta_2) \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.7}$$

On remarque dans 3.7 que les deux dernières colonnes de la matrice de ψ ne dépendent pas de θ_6. On écrit un nouveau système en écrivant l'égalité terme à terme des deux dernières colonnes de 3.7, qui cette fois-ci ne dépend pas de θ_6 :

$$\cos\theta_3 f_1 + \sin\theta_3 f_2 = \cos\theta_2 h_1 + \sin\theta_4 h_2 + r_2 \tag{3.8}$$

$$\sin\theta_3 f_1 - \cos\theta_3 f_2 = -\lambda_2 (\sin\theta_2 h_1 - \cos\theta_4 h_2) + \mu_3 (h_3 - d_2) \tag{3.9}$$

$$f_3 = \mu_2 (\sin\theta_2 h_1 - \cos\theta_4 h_2) + \lambda_2 (h_3 - d_2) \tag{3.10}$$

$$\cos\theta_3 ro_1 + \sin\theta_3 ro_2 = \cos\theta_2 n_1 + \sin\theta_2 h_2 + n_2 \tag{3.11}$$

$$\sin\theta_3 ro_1 - \cos\theta_3 ro_2 = -\lambda_3 (\sin\theta_2 n_1 - \cos\theta_2 n_2) + \mu_3 n_3 \tag{3.12}$$

$$ro_3 = \mu_3 \left(\sin\theta_2 n_1 - \cos\theta_2 n_2\right) + \lambda_3 n_3 \tag{3.13}$$

avec :

$$f_1 = \cos\theta_4 g_1 + \sin\theta_4 g_2 + r_3$$

$$f_2 = -\lambda_3 \left(\sin\theta_4 g_1 - \cos\theta_4 g_2\right) + \mu_3 g_3$$

$$f_3 = \mu_3 \left(\sin\theta_4 g_1 - \cos\theta_4 g_2\right) + \lambda_3 g_3 + d_3$$

$$ro_1 = \cos\theta_4 m_1 + \sin\theta_4 m_2$$

$$ro_2 = -\lambda_3 \left(\sin\theta_4 m_1 - \cos\theta_4 m_2\right) + \mu_3 m_3$$

$$ro_3 = \mu_3 \left(\sin\theta_4 m_1 - \cos\theta_4 m_2\right) + \lambda_3 m_3$$

$$g_1 = \cos\theta_5 r_5 + r_4$$

$$g_2 = -\lambda_4 \sin\theta_5 r_5 + \mu_4 r_4$$

$$g_3 = \mu_4 \sin\theta_5 r_5 + \lambda_4 r_4$$

$$m_1 = \sin\theta_5 \mu_5$$

$$m_2 = \lambda_4 \cos\theta_5 \mu_5 + \mu_4 \lambda_5$$

$$m_3 = -\mu_4 \cos\theta_5 \mu_5 + \lambda_4 \lambda_5$$

$$h_1 = \cos\theta_1 p + \sin\theta_1 q - r_1$$

$$h_2 = -\lambda_1 \left(\sin\theta_1 p - \cos\theta_1 q\right) + \mu_1 \left(r - d_1\right)$$

$$h_3 = \mu_1 \left(\sin\theta_1 p - \cos\theta_1 q\right) + \lambda_1 \left(r - d_1\right)$$

$$n_1 = \cos\theta_1 u + \sin\theta_1 v$$

$$n_2 = -\lambda_1 \left(\sin\theta_1 u - \cos\theta_1 v\right) + \mu_1 w$$

$$n_3 = \mu_1 \left(\sin\theta_1 u - \cos\theta_1 v\right) + \lambda_1 w$$

$$p = -l_x - \left(m_x \mu_6 + n_x \lambda_6\right) d_6 + \rho_x$$

$$q = -l_y - \left(m_y \mu_6 + n_y \lambda_6\right) d_6 + \rho_y$$

$$p = -l_z - \left(m_z \mu_6 + n_z \lambda_6\right) d_6 + \rho_z$$

$$u = m_x \mu_6 + n_x \lambda_6$$

$$v = m_y \mu_6 + n_y \lambda_6$$

$$w = m_z \mu_6 + n_z \lambda_6$$

Grâce aux notation qu'on a introduit en début de section, on va pouvoir réécrire les équations 3.8 à 3.13 de façon beaucoup plus pratique :

$$Rot_2\widetilde{h} = G_2 Rot_3 \widetilde{f} + \widetilde{a_2} \tag{3.14}$$

$$Rot_2\widetilde{n} = G_2 Rot_3 \widetilde{ro} \tag{3.15}$$

Avec $\widetilde{h} = \begin{pmatrix} h_1 \\ h_2 \\ h_3 \end{pmatrix}$, $\widetilde{f} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix}$, $\widetilde{n} = \begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix}$ et $\widetilde{ro} = \begin{pmatrix} ro_1 \\ ro_2 \\ ro_3 \end{pmatrix}$. On note $\widetilde{P_g} = Rot_2\widetilde{h}$, $\widetilde{P_d} = G_2 Rot_3 \widetilde{f} + \widetilde{a_2}$, $\widetilde{l_g} = Rot_2\widetilde{n}$ et enfin $\widetilde{l_d} = G_2 Rot_3 \widetilde{ro}$. On va exploiter les propriétés des composantes des vecteurs $\widetilde{P_g}$, $\widetilde{P_d}$, $\widetilde{l_g}$ et $\widetilde{l_d}$ pour éliminer cinq des six variables du problème. Remarquons tout d'abord que les équations 3.14 et 3.15 sont équivalentes aux équations suivantes :

$$\widetilde{P_g} = \widetilde{P_d} \tag{3.16}$$

et

$$\widetilde{l_g} = \widetilde{l_d} \tag{3.17}$$

Les coordonnées du vecteur $\widetilde{P_g}$ sont des polynômes en les variables $\cos\theta_3, \sin\theta_3, \cos\theta_4, \sin\theta_4, \cos\theta_5$ et $\sin\theta_5$, qu'on regardera ici comme des polynômes de $R[\cos\theta_3, \sin\theta_3][\cos\theta_4, \sin\theta_4, \cos\theta_5, \sin\theta_5]$. Ces polynômes s'expriment tous comme des combinaisons linéaires des monômes : $\sin theta_4 \sin\theta_5, \sin\theta_4 \cos\theta_5, \cos\theta_4 \sin\theta_5, \cos\theta_4 \cos\theta_5, \sin\theta_4, \cos\theta_4, \sin\theta_5, \cos\theta_5, 1$, à coefficients dans $R[\cos\theta_3, \sin\theta_3]$. Il en est de même pour $\widetilde{l_g} \in R[\cos\theta_3, \sin\theta_3][\cos\theta_4, \sin\theta_4, \cos\theta_5, \sin\theta_5]$. Donc la famille

$$F_{4,5} = \{\sin\theta_4 \sin\theta_5, \sin\theta_4 \cos\theta_5, \cos\theta_4 \sin\theta_5, \cos\theta_4 \cos\theta_5, \sin\theta_4, \cos\theta_4, \sin\theta_5, \cos\theta_5, 1\}$$

est une base d'un R-espace vectoriel contenant $\widetilde{P_g}$ et $\widetilde{l_g}$. De la meme façon, la famille

$$F_{1,2} = \{\sin\theta_1 \sin\theta_2, \sin\theta_1 \cos\theta_2, \cos\theta_1 \sin theta_2, \cos\theta_1 \cos\theta_2, \sin\theta_1, \cos\theta_1, \sin\theta_2, \cos\theta_2\}$$

est une base d'un sous-espace vectoriel de $k[\cos\theta_1, \sin\theta_1, \cos\theta_2, \sin\theta_2]$ contenant $\widetilde{P_d}$ et $\widetilde{l_d}$. Donc on peut mettre les équations 3.16 et 3.17 sous la forme suivante :

$$(A) \begin{pmatrix} \sin\theta_4 \sin\theta_5 \\ \sin\theta_4 \cos\theta_5 \\ \cos\theta_4 \sin\theta_5 \\ \cos\theta_4 \cos\theta_5 \\ \sin\theta_4 \\ \cos\theta_4 \\ \sin\theta_5 \\ \cos\theta_5 \\ 1 \end{pmatrix} = (B) \begin{pmatrix} \sin\theta_1 \sin\theta_2 \\ \sin\theta_1 \cos\theta_2 \\ \cos\theta_1 \sin\theta_2 \\ \cos\theta_1 \cos\theta_2 \\ \sin\theta_1 \\ \cos\theta_1 \\ \sin\theta_2 \\ \cos\theta_2 \end{pmatrix} \tag{3.18}$$

Avec $A \in M_{6\times 9}(R)$ et $B \in M_{6\times 8}(R)$. Remarquons bien que les coefficients de $A \in k[\cos\theta_3, \sin\theta_3]$ sont des fonctions de θ_3.

Lemme 1 *$\widetilde{P_g}.\widetilde{P_g}$ appartient au sous-espace vectoriel engendré par la famille $F_{3,4}$ et $\widetilde{P_d}.\widetilde{P_d}$ à celui engendré par $F_{1,2}$.*

La démonstration est donnée en annexe. On se ramène à :

$$\widetilde{P_d}.\widetilde{P_d} = \left(-MG_6\widetilde{a_6} + \widetilde{q}\right).\left(-MG_6\widetilde{a_6} + \widetilde{q}\right) = \left(\widetilde{a_6}.\widetilde{a_6}\right) + \left(\widetilde{q}.\widetilde{q}\right)$$
$$+2\left(\left(-MG_6\widetilde{a_6}\right).\widetilde{q}\right)$$

et

$$\widetilde{P_g}.\widetilde{P_g} = \left(\widetilde{a_5}.\widetilde{a_5}\right) + \left(\widetilde{a_4}.\widetilde{a_4}\right) + \left(\widetilde{a_3}.\widetilde{a_3}\right) + \left(\widetilde{a_2}.\widetilde{a_2}\right) +$$
$$2\left(\left(G_2Rot_3G_3Rot_4G_4Rot_5\widetilde{a_5}\right).\widetilde{a_2}\right) + 2\left(\left(G_2Rot_3G_3Rot_4\widetilde{a_4}\right).\widetilde{a_2}\right) +$$
$$2\left(\left(G_2Rot_3\widetilde{a_3}\right).\widetilde{a_2}\right) + 2\left(\left(G_3Rot_4G_4Rot_5\widetilde{a_5}\right).\widetilde{a_3}\right) + 2\left(\left(G_3Rot_4\widetilde{a_4}\right).\widetilde{a_3}\right) +$$
$$2\left(\left(G_4Rot_5\widetilde{a_5}\right).\widetilde{a_4}\right)$$

Ce sont ces formules qu'on utilise pour le calcul effectif.

On remarque alors que d'après 3.16, on a

$$\widetilde{P_g}.\widetilde{P_g} = \widetilde{P_d}.\widetilde{P_d} \tag{3.19}$$

ce qui nous permet d'obtenir une nouvelle équation. C'est également dans cette optique que nous introduisons les lemmes suivants :

Lemme 2 *$\widetilde{P_g}.\widetilde{l_g}$ appartient au sous-espace vectoriel engendré par la famille $F_{3,4}$ et $\widetilde{P_d}.\widetilde{l_d}$ à celui engendré par $F_{1,2}$.*

On obtient ainsi une nouvelle équation :

$$\widetilde{P_g}.\widetilde{l_g} = \widetilde{P_d}.\widetilde{l_d} \tag{3.20}$$

Lemme 3 *Les coordonnées de $\widetilde{P_g} \times \widetilde{l_g}$ appartient au sous-espace vectoriel engendré par la famille $F_{3,4}$ et celles $\widetilde{P_d} \times \widetilde{l_d}$ à celui engendré par $F_{1,2}$.*

On obtient ainsi trois nouvelles équations par le fait que :

$$\widetilde{P_g} \times \widetilde{l_g} = \widetilde{P_d} \times \widetilde{l_d} \tag{3.21}$$

Lemme 4 *Les coordonnées de $\left(\widetilde{P_g}.\widetilde{P_g}\right)\widetilde{l_g} - 2\left(\widetilde{P_g}.\widetilde{l_g}\right)\widetilde{P_g}$ appartient au sous-espace vectoriel engendré par la famille $F_{3,4}$ et celles $\left(\widetilde{P_d}.\widetilde{P_d}\right)\widetilde{l_d} - 2\left(\widetilde{P_d}.\widetilde{l_d}\right)\widetilde{P_d}$ à celui engendré par $F_{1,2}$.*

On obtient trois nouvelles équations à l'aide de l'égalité :

$$\left(\widetilde{P_g}.\widetilde{P_g}\right)\widetilde{l_g} - 2\left(\widetilde{P_g}.\widetilde{l_g}\right)\widetilde{P_g} = \left(\widetilde{P_d}.\widetilde{P_d}\right)\widetilde{l_d} - 2\left(\widetilde{P_d}.\widetilde{l_d}\right)\widetilde{P_d} \tag{3.22}$$

En résumé, grâce aux égalités 3.16, 3.17, 3.18, 3.20, 3.21 et 3.22, on a un système de quatorze équations, qui par les lemmes 1, 6 et 8 peut être mis sous la forme :

$$(P)\begin{pmatrix} \sin\theta_4\sin\theta_5 \\ \sin\theta_4\cos\theta_5 \\ \cos\theta_4\sin\theta_5 \\ \cos\theta_4\cos\theta_5 \\ \sin\theta_4 \\ \cos\theta_4 \\ \sin\theta_5 \\ \cos\theta_5 \\ 1 \end{pmatrix} = (Q)\begin{pmatrix} \sin\theta_1\sin\theta_2 \\ \sin\theta_1\cos\theta_2 \\ \cos\theta_1\sin\theta_2 \\ \cos\theta_1\cos\theta_2 \\ \sin\theta_1 \\ \cos\theta_1 \\ \sin\theta_2 \\ \cos\theta_2 \end{pmatrix} \tag{3.23}$$

où $P \in M_{14\times 9}(R)$ et $Q \in M_{14\times 8}(R)$, et avec les coefficients de P qui sont des combinaisons linéaires de $\cos\theta_3, \sin\theta_3$ et 1. C'est sur ce système qu'on travaillera désormais pour résoudre le problème. On commence par éliminer θ_1 et θ_2 dans 3.23 dans six des équations. Dans [8], les auteurs utilisent huit des quatorze équations de 3.23 pour éliminer θ_1 et θ_2 dans le membre de gauche. On obtient ainsi avec le membre de droite un système :

$$(\Sigma)\begin{pmatrix} \sin\theta_1\sin\theta_2 \\ \sin\theta_1\cos\theta_2 \\ \cos\theta_1\sin\theta_2 \\ \cos\theta_1\cos\theta_2 \\ \sin\theta_1 \\ \cos\theta_1 \\ \sin\theta_2 \\ \cos\theta_2 \end{pmatrix} = 0 \tag{3.24}$$

où $\Sigma = \Sigma_0 + \sin\theta_3\Sigma_1 + \cos\theta_3\Sigma_2$ est une matrice 6×9. On procède alors aux changements de variables classiques suivant :

$$\sin\theta_3 = \frac{2x_3}{1+x_3^2}, \cos\theta_3 = \frac{1-x_3^2}{1+x_3^2}, \sin\theta_4 = \frac{2x_4}{1+x_4^2}, \cos\theta_4 = \frac{1-x_4^2}{1+x_4^2},$$

$$\sin\theta_5 = \frac{2x_5}{1+x_5^2}, \cos\theta_5 = \frac{1-x_5^2}{1+x_5^2}$$

où $x_i = \tan\left(\frac{\theta_i}{2}\right)$ pour $i = 3, 4, 5$. En multipliant alors 3.24 par $\left(1+x_3^2\right)$,

$\left(1+x_4^2\right)$ et $\left(1+x_5^2\right)$, on est ramené au système suivant :

$$\left(\Sigma'\right)\begin{pmatrix} x_4^2x_5^2 \\ x_4^2x_5 \\ x_4^2 \\ x_4x_5^2 \\ x_4x_5 \\ x_4 \\ x_5^2 \\ x_5 \\ 1 \end{pmatrix} = 0 \tag{3.25}$$

où $\Sigma' \in M_{6\times 9}(R)$ est telle que :

$$\Sigma' = \Sigma'_0 + x_3\Sigma'_1 + x_3^2\Sigma'_2$$

Ce système est à nouveau transformé en utilisant l'élimination dialitique. En particulier si on multiplie l'équation 3.25 par x_4 on obtient un système de la forme :

$$\begin{pmatrix} \Sigma' & 0 \\ 0 & \Sigma' \end{pmatrix}\begin{pmatrix} x_4^3x_5^2 \\ x_4^3x_5 \\ x_4^3 \\ x_4^2x_5^2 \\ x_4^2x_5 \\ x_4^2 \\ x_4x_5^2 \\ x_4x_5 \\ x_4 \\ x_5^2 \\ x_5 \\ 1 \end{pmatrix} = 0 \tag{3.26}$$

où les 0 représentent des blocs de zéros 3×3. Soit

$$\Sigma'' = \begin{pmatrix} \Sigma' & 0 \\ 0 & \Sigma' \end{pmatrix}$$

Alors Σ'' est une matrice 12×12. Le déterminant de cette matrice est un polynôme de degré 24 en x_3. On note $r(x_3)$ ce polynôme. On montre ([8], appendice 3) que ce polynôme est divisible par $\left(1+x_3^2\right)^4$, et ainsi, les angles θ_3 possibles sont solutions d'un polynôme de degré 16. Nous allons maintemant exposer la méthode que nous avons utilisé pour calculer effectivement ces valeurs. Une fois qu'on connaît θ_3, on substitue dans 3.26 pour avoir les valeurs de θ_4 et θ_5, puis on reporte dans 3.23 pour connaître les valeurs de θ_1 et θ_2 et enfin θ_6 sera déterminé par l'équation 3.5.

Chapitre 4

Calcul effectif des solutions

Dans la partie précédente, nous n'avons pas explicité les méthodes qui nous permettent de calculer effectivement les configurations possibles. Différentes méthodes ont déjà été proposées, qui sont soit très proche de la méthode suggérée dans [8], comme dans [3] où on utilise des méthodes numériques de type "décomposition QR ou QZ" pour réduire le système, soit des méthodes qui utilisentes méthodes de cas génériques (qui consistent déformer un système dont on connait les solutions pour obtenir celles du système qu'on a), comme dans [1]. Notre approche présente à la fois l'avantage de la simplicité et celui de la rapidité. Il s'agit d'une solution hybride formelle-numérique inspirée de celle décrite dans [3].

4.1 Partie calcul symbolique

Il s'agit d'utiliser les calculs développés dans l'approche décrite au chapitre précédent. C'est-à-dire qu'on utilise le formalisme introduit. Les équations 3.16 à 3.22 sont calculées avec les formules des preuves des lemmes 1 à 8, sous MAPLE sans affecter de valeur aux paramètres, puis toujours sous MAPLE, à l'aide de la bibliothèque Multires, on extrait les matrices P et Q de l'équation 3.23. Grâce à une extension de la fonction C de MAPLE due aux membres de l'équipe ROBOVIS (INRIA Sophia-Antipolis),on génère quatre matrices C fonctions des angles θ_i. Il y a quatre matrices car la matrice P est décomposée en trois matrices (une qui dépend de $\sin\theta_3$, une qui dépend de $\cos\theta_3$ et une qui est indépendante de θ_3). Ainsi, il suffit dans le code C de rentrer en paramètres la matrice M_{hand}, les r_i, d_i et les α_i pour avoir des matrices de réels, ce qui simplifie le calcul des matrices Σ_0, Σ_1 et Σ_2 qui donnent la matrice Σ de l'équation 3.24. On écrit en fait $P = P_0 + \sin\theta_3 P_1 + \cos\theta_3 P_2$. Une fois ces quatre matrices obtenues, on modifie un peu le code de façon à avoir un structure manipulable par le reste du programme.

4.2 Partie numérique

Comme tout le reste de l'algorithme n'utilise que de la manipulation de matrices, nous avons choisi d'utiliser la bibliothèque ALP, qui est une bibliothèque C++ permettant la manipulation de stucture pour l'algèbre linéaire et polynomiale. Nous avons surtout utilisé le lien entre ALP et LAPACK qui est une bibliothèque FORTRAN de méthodes numériques pour l'algèbre linéaire.

4.2.1 Calcul de la matrice Σ

Pour éliminer les variables θ_1 et θ_2, on applique une décomposition en valeurs singulières à la matrice Q, qui est une matrice de nombre réels.On obtient ainsi trois matrices : $U \in O_{14}(R)$, $V \in O_8(R)$ et $D \in M_{14\times 8}(R)$ telles que $U^tQV = D$. D est une matrice dont tous les termes sont nuls sauf éventuellement les termes de la forme $D_{i,i} \geq 0, i = 1, \ldots, 8$, avec $D_{i,i} \geq D_{j,j}$ si $i > j$. Si le rang de Q est 8, on peut avoir plus de 16 solutions pour chaque instance, si le rang est inférieur à 8, alors on a au plus 16 solutions. C'est ce dernier cas qui nous intéresse. En effet, si le rang est inférieur à 8, on regarde l'équation $U^tQ = DV^t$, et on remarque que le membre de droite a au moins 6 lignes de zero. Alors en multipliant l'équation 3.23 par U^t on obtient :

$$U^tP \begin{pmatrix} \sin\theta_4 \sin\theta_5 \\ \sin\theta_4 \cos\theta_5 \\ \cos\theta_4 \sin\theta_5 \\ \cos\theta_4 \cos\theta_5 \\ \sin\theta_4 \\ \cos\theta_4 \\ \sin\theta_5 \\ \cos\theta_5 \\ 1 \end{pmatrix} = DV^t \begin{pmatrix} \sin\theta_1 \sin\theta_2 \\ \sin\theta_1 \cos\theta_2 \\ \cos\theta_1 \sin\theta_2 \\ \cos\theta_1 \cos\theta_2 \\ \sin\theta_1 \\ \cos\theta_1 \\ \sin\theta_2 \\ \cos\theta_2 \end{pmatrix} \tag{4.1}$$

On considère alors la matrice Σ comme la matrice obtenue en extrayant les 6 premières lignes de U^tP, et on tire de 4.1 l'équation suivante :

$$\Sigma \begin{pmatrix} \sin\theta_4 \sin\theta_5 \\ \sin\theta_4 \cos\theta_5 \\ \cos\theta_4 \sin\theta_5 \\ \cos\theta_4 \cos\theta_5 \\ \sin\theta_4 \\ \cos\theta_4 \\ \sin\theta_5 \\ \cos\theta_5 \\ 1 \end{pmatrix} = 0 \tag{4.2}$$

qui est bien l'équation 3.24. En fait, effectivement on utilise la forme décomposée de P, on calcule U^tP_0, U^tP_1 et U^tP_2 et on extrait les 6 premières lignes pour obtenir 3 nouvelles matrices Σ_0, Σ_1 et Σ_2 telles que $\Sigma_0 + \sin\theta_3\Sigma_1 + \cos\theta_3\Sigma_2 = \Sigma$. Toutes ces matrices sont dans $M_{6\times 9}(R)$. Pour obtenir la matrice Σ' de l'équation 3.25 on constate que Σ ne dépend que de θ_3, et que donc :

$$\left(1 + x_3^2\right)\Sigma = \Sigma_0 + 2x_3\Sigma_1 + \left(1 - x_3^2\right)\Sigma_2$$

Donc $\Sigma' = \Sigma'_0 + x_3^2\Sigma'_1 + x_3^2\Sigma'_2$ où on obtient les matrices Σ'_i de la façon suivante : On note C_j la j-ième colonne de la matrice Σ_i et c'_j celle de la matrice Σ'_i, on procède alors aux affectations suivantes :

$$C'_1 = C_4,\ C'_2 = -2C_3,\ C'_3 = -2C_2,\ C'_4 = -C_4 - C_6,\ C'_5 = 2C_5,$$

$$C'_6 = -C_4 - C_8,\ C'_7 = -C_6 - C_8,\ C'_8 = 2C_3 + 2C_7,\ C'_9 = C_4 + C_6 + C_8 + C_9$$

Pour les changements de variables en $\tan\frac{\theta_4}{2}$ et en $\tan\frac{\theta_5}{2}$, puis pour celui en $\tan\frac{\theta_3}{2}$ on fait :

$$\Sigma'_0 = \Sigma'_0 + \Sigma'_2$$
$$\Sigma'_1 = 2\Sigma'_1$$
$$\Sigma'_2 = -\Sigma'_2$$

La matrice obtenue par élimination dialitique dans l'équation 3.26 est elle aussi donnée par une décomposition $\Sigma'' = \Sigma''_0 + x_3\Sigma''_1 + x_3^2\Sigma''_2$, où

$$\Sigma''_i = \begin{pmatrix} \Sigma'_i & 0 \\ 0 & \Sigma'_i \end{pmatrix}$$

pour $i = 0, 1, 2$. On a donc à résoudre le système suivant :

$$\left(\Sigma''_0 + x_3\Sigma''_1 + x_3^2\Sigma''_2\right)\begin{pmatrix} x_4^3x_5^2 \\ x_4^3x_5 \\ x_4^3 \\ x_4^2x_5^2 \\ x_4^2x_5 \\ x_4^2 \\ x_4x_5^2 \\ x_4x_5 \\ x_4 \\ x_5^2 \\ x_5 \\ 1 \end{pmatrix} = 0 \qquad (4.3)$$

Il s'agit d'un problème de valeurs propres et vecteurs propres généralisé. Une fonction ALP appelant une fonction LAPACK permet de résoudre ce

problème en utilisant la méthode décrite au chapitre 1. Ainsi on détermine les valeurs réelles de θ_3, θ_4 et θ_5 qui sont 2 fois l'arctangente respectif de x_3, x_4 et x_5. En substituant les valeurs trouvées dans l'équation 3.18, on obtient un sytème de la forme :

$$(A) \begin{pmatrix} \sin\theta_4 \sin\theta_5 \\ \sin\theta_4 \cos\theta_5 \\ \cos\theta_4 \sin\theta_5 \\ \cos\theta_4 \cos\theta_5 \\ \sin\theta_4 \\ \cos\theta_4 \\ \sin\theta_5 \\ \cos\theta_5 \\ 1 \end{pmatrix} = b \tag{4.4}$$

qui nous donne les valeurs de θ_1 et θ_2. On détermine alors θ_6 aux égalités terme à terme dans 3.7.

Bibliographie

[1] Alexander Morgan Charles Wampler. Solving the 6r inverse position problem using a generic-case solution methodology. *Mechanism and Machine Theory*, 26(1) :91–106, 1991.

[2] Donald O'Shea David Cox, John Little. *Ideals, varieties and algorithm.* Springer-Verlag, 1996.

[3] John F. Canny Dinesh Manocha. Real time inverse kinematics for general 6r manipulator. In *International Conference on Robotics and Automations*, 1992. Nice.

[4] D. Eisenbud. *Commutative algebra with a view towards algebraic geometry.* Springer-Verlag, 1994.

[5] KUANGRONG HAO. Dual number, lie algebra and 6r inverse kinematics. In *Advance in robot kinematics*, 1996.

[6] Bernard Mourrain Ioannis Z. Emiris. Matrix in elimination theory. *Journal of symbolic computation*, 1999.

[7] S. Zeghloul J.-P. Lallemand. *Robotique.* Masson.

[8] Bernard Roth Madshusudan Raghavan. Kinematics analysis of the 6r manipulator of general geometry. In *International Symposium on Robotics Research*, pages 314–320, 1989. Tokyo.

[9] Bernard Mourrain. An introduction to linear algebra methods for solving polynomial equations. http ://www-sop.inria.fr/saga/mourrain/publi.html.

[10] Charles R. Johnson Roger A. Horn. *Topics in matrix analysis.* Cambridge University Press.

[11] I. R. Shafarevitch. *Basic Algebraic Geometry.* Springer-Verlag, 1974.

[12] Bernd Sturmfels. Introduction to resultants. http ://math.berkeley.edu/ bernd/papers/resultants.ps.

.1 Preuves des lemmes 1, 2, 3 et 4

Lemme 5 *$\widetilde{P_g}.\widetilde{P_g}$ appartient au sous-espace vectoriel engendré par la famille $F_{3,4}$ et $\widetilde{P_d}.\widetilde{P_d}$ à celui engendré par $F_{1,2}$.*

Preuve

$$\widetilde{P_g}.\widetilde{P_g} = \left(G_2 Rot_3 \widetilde{f} + \widetilde{a_2}\right).\left(G_2 Rot_3 \widetilde{f} + \widetilde{a_2}\right)$$

D'où

$$\widetilde{P_g}.\widetilde{P_g} = \left(\widetilde{f}.\widetilde{f}\right) + (\widetilde{a_2}.\widetilde{a_2}) + 2\left(\left(G_2 Rot_3 \widetilde{f}\right).\widetilde{a_2}\right)$$

Comme $\widetilde{f} = G_3 Rot_4 \widetilde{g} + \widetilde{a_3}$, on obtient :

$$\widetilde{P_g}.\widetilde{P_g} = (\widetilde{g}.\widetilde{g}) + 2\left((G_3 Rot_4 \widetilde{g}).\widetilde{a_3}\right) + 2\left((G_2 Rot_3 G_3 Rot_4 \widetilde{g}).\widetilde{a_2}\right) + 2\left((G_2 Rot_3 \widetilde{a_3}).\widetilde{a_2}\right) + (\widetilde{a_3}.\widetilde{a_3}) + (\widetilde{a_2}.\widetilde{a_2})$$

Puis comme $\widetilde{g} = G_4 Rot_5 \widetilde{a_5} + \widetilde{a_4}$, on obtient finalement :

$$\widetilde{P_g}.\widetilde{P_g} = (\widetilde{a_5}.\widetilde{a_5}) + (\widetilde{a_4}.\widetilde{a_4}) + (\widetilde{a_3}.\widetilde{a_3}) + (\widetilde{a_2}.\widetilde{a_2}) + 2\left((G_2 Rot_3 G_3 Rot_4 G_4 Rot_5 \widetilde{a_5}).\widetilde{a_2}\right) + 2\left((G_2 Rot_3 G_3 Rot_4 \widetilde{a_4}).\widetilde{a_2}\right) + 2\left((G_2 Rot_3 \widetilde{a_3}).\widetilde{a_2}\right) + 2\left((G_3 Rot_4 G_4 Rot_5 \widetilde{a_5}).\widetilde{a_3}\right) + 2\left((G_3 Rot_4 \widetilde{a_4}).\widetilde{a_3}\right) + 2\left((G_4 Rot_5 \widetilde{a_5}).\widetilde{a_4}\right)$$

Sous cette forme il est clair que $\widetilde{P_g}$ est bien dans le sous-espace vectoriel sur $R[\cos\theta_3, \sin\theta_3]$ engendré par la famille $F_{4,5}$. Intéressons-nous maintenant à $\widetilde{P_d}$:

$$\widetilde{P_d}.\widetilde{P_d} = \left(Rot_2 \widetilde{h}\right).\left(Rot_2 \widetilde{h}\right) = \widetilde{h}.\widetilde{h}$$

Or $\widetilde{h} = G_1 Rot_1 \widetilde{v}$, donc :

$$\widetilde{P_d}.\widetilde{P_d} = \left(G_1 Rot_1 \widetilde{v}\right).\left(G_1 Rot_1 \widetilde{v}\right) = \widetilde{v}.\widetilde{v}$$

Comme $\widetilde{v} = -MG_6 \widetilde{a_6} + \widetilde{q}$, on a :

$$\widetilde{P_d}.\widetilde{P_d} = \left(-MG_6 \widetilde{a_6} + \widetilde{q}\right).\left(-MG_6 \widetilde{a_6} + \widetilde{q}\right) = (\widetilde{a_6}.\widetilde{a_6}) + (\widetilde{q}.\widetilde{q}) + 2\left((-MG_6 \widetilde{a_6}).\widetilde{q}\right)$$

Là encore, il est facile de constater que $\widetilde{P_d}.\widetilde{P_d}$ est bien dans le sous-espace

vectoriel sur R engendré par la famille $F_{1,2}$. Ce qui démontre le lemme 1. ♣

Lemme 6 *$\widetilde{P_g}.\widetilde{l_g}$ appartient au sous-espace vectoriel engendré par la famille $F_{3,4}$ et $\widetilde{P_d}.\widetilde{l_d}$ à celui engendré par $F_{1,2}$.*

Preuve

$$\widetilde{P_g}.\widetilde{l_g} = \widetilde{l_g}.\widetilde{P_g} = (G_2 Rot_3 \widetilde{ro}) \,.\, \left(G_2 Rot_3 \widetilde{f} + \widetilde{a_2}\right)$$

Et comme $\widetilde{ro} = G_3 Rot_4 \widetilde{m}$ et que $\widetilde{f} = G_3 Rot_4 \widetilde{g} + \widetilde{a_3}$, on a :

$$\widetilde{P_g}.\widetilde{l_g} = (G_2 Rot_3 G_3 Rot_4 \widetilde{m}) \,.\, (G_2 Rot_3 \,(G_3 Rot_4 \widetilde{g} + \widetilde{a_3}) + \widetilde{a_2})$$

Or $\widetilde{m} = G_4 Rot_5 G_5 \widetilde{e}$ et $\widetilde{g} = G_4 Rot_5 \widetilde{a_5} + \widetilde{a_4}$, on a finalement :

$$\begin{gathered}\widetilde{P_g}.\widetilde{l_g} = ((G_5\widetilde{e})\,.\widetilde{a_5}) + ((G_4 Rot_5 G_5 \widetilde{e})\,.\widetilde{a_4}) + \\ ((G_3 Rot_4 G_4 Rot_5 G_5 \widetilde{e})\,.\widetilde{a_3}) + \\ ((G_2 Rot_3 G_3 Rot_4 G_4 Rot_5 G_5 \widetilde{e})\,.\widetilde{a_2})\end{gathered}$$

qui s'exprime bien dans la base monômiale souhaitée. De même :

$$\widetilde{P_d}.\widetilde{l_d} = \left(Rot_2 \widetilde{h}\right) \,.\, (Rot_2 \widetilde{n}) = \widetilde{h}.\widetilde{n}$$

Or $\widetilde{h} = G_1 Rot_1 \widetilde{v}$ et $\widetilde{n} = G_1 Rot_1 \widetilde{u}$, donc :

$$\widetilde{P_d}.\widetilde{l_d} = \widetilde{v}.\widetilde{u} = \widetilde{q}.\,(M G_6 \widetilde{e}) - \widetilde{a_6}.\widetilde{e}$$

Ce qui suffit à voir qu'on est bien dans la situation du lemme.♣

Lemme 7 *Les coordonnées de $\widetilde{P_g} \times \widetilde{l_g}$ appartient au sous-espace vectoriel engendré par la famille $F_{3,4}$ et celles $\widetilde{P_d} \times \widetilde{l_d}$ à celui engendré par $F_{1,2}$.*

Preuve Un simple calcul montre que :

$$\begin{gathered}\widetilde{P_g} \times \widetilde{l_g} = (\widetilde{a_5} \times (G_5\widetilde{e})) + (\widetilde{a_4} \times (G_4 Rot_5 G_5 \widetilde{e})) + \\ (\widetilde{a_3} \times (G_3 Rot_4 G_4 Rot_5 G_5 \widetilde{e})) + \\ (\widetilde{a_2} \times (G_2 Rot_3 G_3 Rot_4 G_4 Rot_5 G_5 \widetilde{e}))\end{gathered}$$

De même on a facilement :

$$\widetilde{P_d} \times \widetilde{l_d} = Rot_2 G_1 Rot_1 M G_6 \,(-\widetilde{a_6} \times \widetilde{e}) + Rot_2 G_1 Rot_1 \,(\widetilde{q} \times M G_6 \widetilde{e})$$

Avec lequel on voit facilement le résultat.♣

Lemme 8 *Les coordonnées de* $\left(\widetilde{P_g}.\widetilde{P_g}\right)\widetilde{l_g}-2\left(\widetilde{P_g}.\widetilde{l_g}\right)\widetilde{P_g}$ *appartient au sous-espace vectoriel engendré par la famille* $F_{3,4}$ *et celles* $\left(\widetilde{P_d}.\widetilde{P_d}\right)\widetilde{l_d}-2\left(\widetilde{P_d}.\widetilde{l_d}\right)\widetilde{P_d}$ *à celui engendré par* $F_{1,2}$.

Preuve Dans un premièr temps intéressons-nous à $\left(\widetilde{P_g}.\widetilde{P_g}\right)\widetilde{l_g}-2\left(\widetilde{P_g}.\widetilde{l_g}\right)\widetilde{P_g}$:

$$\left(\widetilde{P_g}.\widetilde{P_g}\right)\widetilde{l_g}-2\left(\widetilde{P_g}.\widetilde{l_g}\right)\widetilde{P_g}=\left(\widetilde{f}.\widetilde{f}\right)G_2Rot_3\widetilde{ro}+2\left(G_2Rot_3\widetilde{f}.\widetilde{a_2}\right)G_2Rot_3\widetilde{ro}$$
$$-2\left(\widetilde{f}.\widetilde{ro}\right)G_2Rot_3\widetilde{f}-2\left(\widetilde{f}.\widetilde{ro}\right)\widetilde{a_2}-2\left(\widetilde{a_2}.G_2Rot_3\widetilde{ro}\right)G_2Rot_3\widetilde{f}$$
$$-2\left(\widetilde{a_2}.G_2Rot_3\widetilde{ro}\right)\widetilde{a_2}$$

D'où :

$$\left(\widetilde{P_g}.\widetilde{P_g}\right)\widetilde{l_g}-2\left(\widetilde{P_g}.\widetilde{l_g}\right)\widetilde{P_g}=G_2Rot_3\left\{\left(\widetilde{f}.\widetilde{f}\right)\widetilde{ro}-2\left(\widetilde{f}.\widetilde{ro}\right)\widetilde{f}+\left(\widetilde{a_2}.\widetilde{a_2}\right)\widetilde{ro}\right\}$$
$$-2\left(\widetilde{f}.\widetilde{ro}\right)\widetilde{a_2}-2\left(\widetilde{a_2}.G_2Rot_3\widetilde{ro}\right)\widetilde{a_2}+2\left(G_2Rot_3\left(\widetilde{f}\times\widetilde{ro}\right)\right)\times\widetilde{a_2}$$

Car :

$$2\left(G_2Rot_3\widetilde{f}.\widetilde{a_2}\right)G_2Rot_3\widetilde{ro}-2\left(\widetilde{a_2}.G_2Rot_3\widetilde{ro}\right)G_2Rot_3\widetilde{f}=2\left(G_2Rot_3\left(\widetilde{f}\times\widetilde{ro}\right)\right)\times\widetilde{a_2}$$

Comme :

$$\left(\widetilde{f}.\widetilde{f}\right)\widetilde{ro}-2\left(\widetilde{f}.\widetilde{ro}\right)\widetilde{f}=G_3Rot_4\left\{\left(\widetilde{g}.\widetilde{g}\right)\widetilde{m}-2\left(\widetilde{g}.\widetilde{m}\right)\widetilde{g}+\left(\widetilde{a_3}.\widetilde{a_3}\right)\widetilde{m}\right\}$$
$$-2\left(\widetilde{g}.\widetilde{m}\right)\widetilde{a_3}-2\left(\widetilde{a_3}.G_3Rot_4\widetilde{m}\right)\widetilde{a_3}+$$
$$\left(G_3Rot_4\left(\widetilde{g}\times\widetilde{m}\right)\right)\times\widetilde{a_3}$$

Puis on constate que :

$$\left(\widetilde{g}.\widetilde{g}\right)\widetilde{m}-2\left(\widetilde{g}.\widetilde{m}\right)\widetilde{g}=G_4Rot_5\left\{\left(\widetilde{a_5}.\widetilde{a_5}\right)G_5\widetilde{e}+\left(\widetilde{a_4}.\widetilde{a_4}\right)G_5\widetilde{e}+\left(\widetilde{a_5}.G_5\widetilde{e}\right)\widetilde{a_5}\right\}$$
$$-2\left\{\left(\widetilde{a_5}.G_5\widetilde{e}\right)+\left(G_4Rot_5G_5\widetilde{e}\right)\right\}\widetilde{a_4}+\left(G_4Rot_5\left(\widetilde{a_5}\times G_5\widetilde{e}\right)\right)\times\widetilde{a_4}$$

Ce qui nous amène à écrire :

$$\left(\widetilde{P_g}.\widetilde{P_g}\right)\widetilde{l_g}-2\left(\widetilde{P_g}.\widetilde{l_g}\right)\widetilde{P_g}=G_2Rot_3\left\{G_3Rot_4\left(G_4Rot_5A+B\right)+C\right\}+D$$

Avec :

$$A = (\widetilde{a_5}.\widetilde{a_5})\, G_5\widetilde{e} + (\widetilde{a_4}.\widetilde{a_4})\, G_5\widetilde{e} + (\widetilde{a_5}.G_5\widetilde{e})\, \widetilde{a_5}$$

$$B = -2\,(\widetilde{a_5}.G_5\widetilde{e})\, \widetilde{a_4} - 2\,(\widetilde{a_4}.G_4Rot_5G_5\widetilde{e})\, \widetilde{a_4} + (G_4Rot_5\,(\widetilde{a_5} \times G_5\widetilde{e})) \times \widetilde{a_4}$$

$$C = -2\,(\widetilde{a_5}.G_5\widetilde{e})\, \widetilde{a_3} - 2\,(\widetilde{a_4}.G_4Rot_5G_5)\, \widetilde{a_3}$$

$$-2\,(\widetilde{a_3}.G_3Rot_4G_4Rot_5G_5)\, \widetilde{a_3} + (G_3Rot_4\,(\widetilde{a_4} \times G_4Rot_5G_5\widetilde{e})) \times \widetilde{a_3} +$$

$$(G_3Rot_4G_4Rot_5\,(\widetilde{a_5} \times G_5\widetilde{e})) \times \widetilde{a_3}$$

$$D = -2\,(\widetilde{a_5}.G_5\widetilde{e})\, \widetilde{a_2} - 2\,(\widetilde{a_4}.G_4Rot_5G_5\widetilde{e})\, \widetilde{a_2} - 2\,(\widetilde{a_3}.G_3Rot_4G_4Rot_5G_5\widetilde{e})$$

$$+ (G_2Rot_3G_3Rot_4G_4Rot_5\,(\widetilde{a_5} \times G_5\widetilde{e})) \times \widetilde{a_2} +$$

$$(G_2Rot_3G_3Rot_4\,(\widetilde{a_4} \times G_4Rot_5G_5\widetilde{e})) \times \widetilde{a_2}$$

$$+ (G_2Rot_3\,(\widetilde{a_4} \times G_3Rot_4G_4Rot_5G_5\widetilde{e}))$$

$$-2\,(\widetilde{a_2}.\,(G_2Rot_3G_3Rot_4G_4Rot_5G_5\widetilde{e}))$$

Sous cette forme il est facile de vérifier la première affirmation du lemme. Passons donc à la deuxième affirmation :

$$\left(\widetilde{P_d}.\widetilde{P_d}\right) \widetilde{l_d} - 2\left(\widetilde{P_d}.\widetilde{l_d}\right) \widetilde{P_d} = \left(\widetilde{h}.\widetilde{h}\right) Rot_2\widetilde{n} - 2\left(\widetilde{h}.\widetilde{n}\right) Rot_2\widetilde{h}$$

D'où :

$$\left(\widetilde{P_d}.\widetilde{P_d}\right) \widetilde{l_d} - 2\left(\widetilde{P_d}.\widetilde{l_d}\right) \widetilde{P_d} = Rot_2\,\{(\widetilde{v}.\widetilde{v})\, G_1Rot_1\widetilde{u} - 2\,(\widetilde{v}.\widetilde{u})\, G_1Rot_1\widetilde{v}\}$$

Soit :

$$\left(\widetilde{P_d}.\widetilde{P_d}\right) \widetilde{l_d} - 2\left(\widetilde{P_d}.\widetilde{l_d}\right) \widetilde{P_d} = Rot_2G_1Rot_1\,\{(\widetilde{a_6}.\widetilde{a_6}) + (\widetilde{q}.\widetilde{q}) - 2\,(MG_6\widetilde{a_6}.\widetilde{q})$$

$$+2\,(\widetilde{a_6}.\widetilde{e})\}\, MG_6\widetilde{e} + 2\,((\widetilde{a_6}.\widetilde{e}) - (\widetilde{q}.MG_6\widetilde{e}))$$

D'où le résultat.♣

.2 Le code MAPLE

Ce code sert à générer le code C des matrices A et B de 3.18. Ce code est modifié pour être manipulable à partir de ALP.

```
restart:
libname:="/u/corse/2/robovis/maple/lib5.5",libname:
with(linalg):
readlib(C):
read "prog/multires.mpl":



#*******definition des vecteurs de parametres********
#lambda et mu correspondent aux vecteurs des cosinus et
#sinus des angles alpha
lambda:=array(0..5):
mu:=array(0..5):

# a est le vecteurs des translations selon les axes x etd selon z
a:=array(0..5):
d:=array(0..5):

# s et s sont les sin et cos des angles que nous cherchons (theta)
s:=array(0..5):
c:=array(0..5):

# M est la matrice de chgt de la main vers la premiere liaison
Hand = array(0..3,0..3):



#******definition de variables simplifiant le calcul ***
for i from 1 to 6 do
# les rotation autours des axes x
G[i]:=matrix(3,3,[1, 0, 0, 0, lambda[i-1], mu[i-1], 0, -mu[i-1],
 lambda[i-1]]):
#les rotations autour des axes z
R[i]:=matrix(3,3,[c[i-1], s[i-1], 0, -s[i-1], c[i-1], 0, 0, 0, 1]):
# les vects de translation
a0d[i]:=vector(3,[a[i-1], 0, d[i-1]]):
od:
ee:=vector(3,[0, 0, 1]):
q:=vector(3,[Hand[0,3],Hand[1,3],Hand[2,3]]):
```

```
M:=matrix(3,3,[Hand[0,0],Hand[0,1],Hand[0,2],Hand[1,2],Hand[1,1],
Hand[1,2],Hand[2,0],Hand[2,1],Hand[2,2]]):


#DEFINITION DU MEMBRE DE GAUCHE

pg:=map(expand,evalm(G[2]&*R[3]&*evalm(G[3]&*R[4]&*evalm(G[4]&*R[5]&*a0d[5]
+a0d[4])+a0d[3])+a0d[2])):

ppg:=innerprod(evalm(2*evalm(G[4]&*R[5]&*a0d[5])),a0d[4])
+innerprod(evalm(2*evalm(G[3]&*R[4]&*G[4]&*R[5]&*a0d[5])),a0d[3])
+innerprod(evalm(2*evalm(G[3]&*R[4]&*a0d[4])),a0d[3])
+innerprod(evalm(2*evalm(G[2]&*R[3]&*G[3]&*R[4]&*G[4]&*R[5]&*a0d[5])),
a0d[2])+innerprod(evalm(2*evalm(G[2]&*R[3]&*G[3]&*R[4]&*a0d[4])),a0d[2])
+innerprod(evalm(2*evalm(G[2]&*R[3]&*a0d[3])),a0d[2])
+innerprod(a0d[5]*a0d[5])+innerprod(a0d[4],a0d[4])
+innerprod(a0d[3],a0d[3])+innerprod(a0d[2]*a0d[2]):

plg:=innerprod(evalm(G[5]&*ee),a0d[5])+
innerprod(evalm(G[4]&*R[5]&*G[5]&*ee),a0d[4])+
innerprod(evalm(G[3]&*R[4]&*G[4]&*R[5]&*G[5]&*ee),a0d[3])
+innerprod(evalm(G[2]&*R[3]&*G[3]&*R[4]&*G[4]&*R[5]&*G[5]&*ee),a0d[2]):

pxlg:=map(expand,evalm(crossprod(a0d[5],evalm(G[5]&*ee))
+crossprod(a0d[4],evalm(G[4]&*R[5]&*G[5]&*ee))+
crossprod(a0d[3],evalm(G[3]&*R[4]&*G[4]&*R[5]&*G[5]&*ee))+
crossprod(a0d[2],evalm(G[2]&*R[3]&*G[3]&*R[4]&*G[4]&*R[5]&*G[5]&*ee)))):

#un cas un peu plus dur: calcul de (p.p)*l-2(p.l)*p a gauche

A:=map(expand,evalm(evalm(evalm(innerprod(a0d[5],a0d[5])
+innerprod(a0d[5],a0d[5]))*evalm(G[5]&*ee))
+evalm(innerprod(a0d[5],evalm(G[5]&*ee))*a0d[4]))):


B:=map(expand,evalm(evalm(-2*innerprod(a0d[5],evalm(G[5]&*ee))*a0d[4])
-evalm(2*innerprod(a0d[4],evalm(G[4]&*R[5]&*G[5]&*ee))*a0d[4])
-evalm(crossprod(evalm(G[4]&*R[5]&*
crossprod(a0d[5],evalm(G[5]&*ee))),a0d[4])))):


C1:=map(expand,evalm(-2*evalm(innerprod(a0d[5],evalm(G[5]&*ee))
+innerprod(a0d[4],evalm(G[4]&*R[5]&*G[5]&*ee))+
innerprod(a0d[3],evalm(G[3]&*R[4]&*G[4]&*R[5]&*G[5]&*ee)))*a0d[3])):
```

```
C2:=map(expand,evalm(crossprod(evalm(G[3]&*R[4]&*G[4]&*R[5]&*
crossprod(a0d[5],evalm(G[5]&*ee))),a0d[3])+
crossprod(evalm(G[3]&*R[4]&*crossprod(a0d[4],
evalm(G[4]&*R[5]&*G[5]&*ee))),a0d[3]))):

C:=map(expand,evalm(C1+C2)):


D1:=map(expand,evalm(-2*evalm(innerprod(a0d[5],evalm(G[5]&*ee))
+innerprod(a0d[4],evalm(G[4]&*R[5]&*G[5]&*ee))+innerprod(a0d[3],
evalm(G[3]&*R[4]&*G[4]&*R[5]&*G[5]&*ee))+innerprod(a0d[2],
evalm(G[2]&*R[3]&*G[3]&*R[4]&*G[4]&*R[5]&*G[5]&*ee))
+innerprod(a0d[2],
evalm(G[2]&*R[3]&*G[3]&*R[4]&*G[4]&*R[5]&*G[5]&*ee)))*a0d[2])):

D2:=map(expand,evalm(crossprod(evalm(G[2]&*R[3]&*G[3]&*R[4]&*G[4]&*R[5]&*
crossprod(a0d[5],evalm(G[5]&*ee))),a0d[2])
+evalm(G[2]&*R[3]&*G[3]&*R[4]&*crossprod(a0d[4],
evalm(G[4]&*R[5]&*G[5]&*ee)))+evalm(G[2]&*R[3]&*crossprod(a0d[3],
evalm(G[3]&*R[4]&*G[4]&*R[5]&*G[5]&*ee))))):

DD:=map(expand,evalm(D1+D2)):

#resultat
pplg:=map(expand,evalm(evalm(G[4]&*R[5]&*A)+B)):
pplg:=map(expand,evalm(evalm(G[3]&*R[4]&*pplg)+C)):
pplg:=map(expand,evalm(evalm(G[2]&*R[3]&*pplg)+DD)):



#DEFINITION DU MEMBRE DE DROITE

pd:=map(expand,evalm(R[2]&*G[1]&*R[1]&*evalm(q-evalm(M&*G[6]&*a0d[6])))):

ppd:=expand(evalm(innerprod(a0d[6],a0d[6])+innerprod(q,q)
-evalm(2*innerprod(evalm(M&*G[6]&*a0d[6]),q)))):

pld:=expand(innerprod(q,evalm(M&*G[6]&*ee))-innerprod(a0d[6],ee)):

pxld:=map(expand,evalm(evalm(R[2]&*G[1]&*R[1]&*
crossprod(q,evalm(M&*G[6]&*ee)))-
evalm(R[2]&*G[1]&*R[1]&*M&*crossprod(a0d[6],ee)))):
```

```
ppld:=map(expand,evalm(R[2]&*G[1]&*R[1]&*
evalm(expand(innerprod(a0d[6],a0d[6])+innerprod(q,q)
-2*innerprod(evalm(M&*G[6]&*a0d[6]),q)
+2*innerprod(a0d[6],ee))*evalm(M&*G[6]&*ee)
+ evalm(2*innerprod(a0d[6],ee)+2*innerprod(q,evalm(M&*G[6]&*ee)))
+ evalm(2*innerprod(q,evalm(M&*G[6]&*ee))))):

#*************************
# CONSTRUCTION DES MATRICES

GAUCHE := matrixof([pg[1],pg[2],pg[3],ppg,plg,pxlg[1],pxlg[2],
pxlg[3],pplg[1],pplg[2],pplg[3]],[s[3]*s[4],s[3]*c[4],
c[3]*s[4],c[3]*c[4],s[3],c[3],s[4],c[4],1]):

DROITE := matrixof([pd[1],pd[2],pd[3],ppd,pld,pxld[1],pxld[2],
pxld[3],ppld[1],ppld[2],ppld[3]],[s[0]*s[1],s[0]*c[1],
c[0]*s[1],c[0]*c[1],s[0],c[0],s[1],c[1]]):



Q1:=transpose(DROITE):
P1:=transpose(GAUCHE):

Q2:=convert(evalm(Q1),listlist):
Q3:=map(op,%):
Nom := `matun`:
rv := ["double *R"]:
param := ["double *lambda","double *mu","double *a","double *d",
"double *c","double *s"]:
glob := []:
loc:=[]:
ass := [seq(R[i-1] = Q3[i],i=1..nops(Q3))]:
pb :=ProgramBlock(loc,ass):
np := NamedProgram(Nom,rv,param,glob,pb):
C(np,filename = "prog2/mat1.c",optimized):



P2:=subs(c[3]=0,s[3]=0,evalm(P1)):
P3:=convert(evalm(P2),listlist):
P4:=map(op,%):
Nom := `matdeux`:
rv := ["double *O"]:
param := ["double *lambda","double *mu","double *a","double *d"]:
```

```
glob := []:
loc:=[]:
ass := [seq(O[i-1] = P4[i],i=1..nops(P4))]:
pb :=ProgramBlock(loc,ass):
np := NamedProgram(Nom,rv,param,glob,pb):
C(np,filename = "prog2/mat2.c",optimized):


P5:=subs(c[3]=1,s[3]=0,evalm(P1)):
P5:=evalm(P5-P2):
P6:=convert(evalm(P5),listlist):
P7:=map(op,%):
Nom := `mattrois`:
rv := ["double *O"]:
param := ["double *lambda","double *mu","double *a","double *d"]:
glob := []:
loc:=[]:
ass := [seq(O[i-1] = P7[i],i=1..nops(P7))]:
pb :=ProgramBlock(loc,ass):
np := NamedProgram(Nom,rv,param,glob,pb):
C(np,filename = "prog2/mat3.c",optimized):



P8:=subs(c[3]=0,s[3]=1,evalm(P1)):
P8:=evalm(P8-P2):
P9:=convert(evalm(P8),listlist):
P10:=map(op,%):
Nom := `matquat`:
rv := ["double *O"]:
param := ["double *lambda","double *mu","double *a","double *d"]:
glob := []:
loc:=[]:
ass := [seq(O[i-1] = P10[i],i=1..nops(P10))]:
pb :=ProgramBlock(loc,ass):
np := NamedProgram(Nom,rv,param,glob,pb):
C(np,filename = "prog2/mat4.c",optimized):
```

.3 Le code ALP

.3.1 Fichier principal

```
#include "math.h"
#include "_Matrix.H"
#include "linear/VectStd.H"
#include "linear/Svd.H"
#include "linear/Eigen.H"
#include "affect.H"
#include "mymat.H"

int main(int argc, char **argv)
{
  /*declaration des vecteurd de parametres */
  double l[]={0.8732624548, -0.1253332336, -0.3268880288,
-0.9226727398,
 0.9886517449, -0.9716317327};
  double m[]={0.4872501257, 0.9921147013, -0.9450630755,
-0.3855839923,
 -0.1502255879, -0.2364989977};
  double at[]={57, 60, 74, 85, 16, 61};
  double dt[]={27, 69, 6, 78, 26, 29};
  double mt[]={-0.3038864838, -0.0558734824, 0.9510684302,
100.0606135,
 -0.2993526082, -0.9421188150, -0.1509972007, -66.54189987,
0.9044562018,
 -0.3305908234, 0.2695713009, 140.8290988, 0, 0, 0, 1};
  int r = 0;
  int i,j;

  /* On donne les entrees de l'algo*/
  TAB lambda(6,l);
  TAB mu(6,m);
  TAB a(6,at);
  TAB d(6,dt);
  MAT MM(3,4,mt);
  STDLAP Q(14,8),U(14,14),V(8,8),P1(14,9),P2(14,9),P3(14,9);


  /*initialisation des matrices avec lesquels on travail */
  affectmatun(Q.rep, lambda, mu, a, d, MM);
  cout<<"Matrice Q affectee"<<endl;
```

```
 /*on applique une svd a Q */
 Svd(Q, U, V);

 /*Calcul du rang de la matrice Q */
 r = Rank(Q);
 cout<<"Voici le rang de Q"<<r<<endl;

 /* Initialisation de P1,P2 et P3 */
 affectmatdeux(P1.rep, lambda, mu, a, d);
 cout<<"Matrice P1 affectee"<<endl;

 affectmattrois(P2.rep, lambda, mu, a, d);
 cout<<"Matrice P2 affectee"<<endl;

 affectmatquat(P3.rep, lambda, mu, a, d);
 cout<<"Matrice P3 affectee"<<endl;

 /* On transpose la matrice U */
 U.transpose();
 cout<<"Matrice U transposee"<<endl;

 /* On multiplie P1, P2 et P3 par la matrice U */
 P1 *= U;
 P2 *= U;
 P3 *= U;

 /* On extrait les lignes correspondantes aux lignes nuls de U*Q */
 STDLAP P1p(9,6), P2p(9,6), P3p(9,6);
 for(i = 0; i < 6; i++ ){
   for(j = 0; j < 9; j++){
     P1p(i,j) = P1(i + 8, j);
     P2p(i,j) = P2(i + 8, j);
     P3p(i,j) = P3(i + 8, j);
   }
 }
 cout<<"Extraction apres elimination de deux variables
effectuees"<<endl;

 STDLAP Sig0p(6,9), Sig1p(6,9), Sig2p(6,9);
 for(int i = 0; i < 6; i++)
   {
     Sig0p(i,1) = P1p(i,4);
     Sig1p(i,1) = P2p(i,4);
     Sig2p(i,1) = P3p(i,4);
```

```
      Sig0p(i,2) = -2*P1p(i,3);
      Sig1p(i,2) = -2*P2p(i,3);
      Sig2p(i,2) = -2*P3p(i,3);
      Sig0p(i,3) = -2*P1p(i,2);
      Sig1p(i,3) = -2*P2p(i,2);
      Sig2p(i,3) = -2*P3p(i,2);
      Sig0p(i,4) = P1p(i,1);
      Sig1p(i,4) = P2p(i,1);
      Sig2p(i,4) = P3p(i,1);
      Sig0p(i,5) = -P1p(i,4)-P1p(i,6);
      Sig1p(i,5) = -P2p(i,4)-P2p(i,6);
      Sig2p(i,5) = -P3p(i,4)-P3p(i,6);
      Sig0p(i,6) = P1p(i,5);
      Sig1p(i,6) = P2p(i,5);
      Sig2p(i,6) = P3p(i,5);
      Sig0p(i,7) = -P1p(i,4)-P1p(i,8);
      Sig1p(i,7) = -P2p(i,4)-P2p(i,8);
      Sig2p(i,7) = -P3p(i,4)-P3p(i,8);
      Sig0p(i,8) = P1p(i,7)+P1p(i,3);
      Sig1p(i,8) = P2p(i,7)+P2p(i,3);
      Sig2p(i,8) = P3p(i,7)+P3p(i,3);
      Sig0p(i,9) = P1p(i,4)+P1p(i,6)+P1p(i,8)+P1p(i,9);
      Sig1p(i,9) = P2p(i,4)+P2p(i,6)+P2p(i,8)+P2p(i,9);
      Sig2p(i,9) = P3p(i,4)+P3p(i,6)+P3p(i,8)+P3p(i,9);
    }
  /*definition de la matrice sigma' de l'article*/
  STDLAP zero(3,3);
  Init_STDLAP(zero);
  STDLAP Sig0s(12,12), Sig1s(12,12), Sig2s(12,12);
  SetSubMatrix(Sig0s, 0, 0, Sig0p);
  SetSubMatrix(Sig0s, 0, 6, zero);
  SetSubMatrix(Sig0s, 6, 0, zero);
  SetSubMatrix(Sig0s, 6, 3, Sig0p);
  SetSubMatrix(Sig1s, 0, 0, Sig1p);
  SetSubMatrix(Sig1s, 0, 6, zero);
  SetSubMatrix(Sig1s, 6, 0, zero);
  SetSubMatrix(Sig1s, 6, 3, Sig1p);
  SetSubMatrix(Sig2s, 0, 0, Sig2p);
  SetSubMatrix(Sig2s, 0, 6, zero);
  SetSubMatrix(Sig2s, 6, 0, zero);
  SetSubMatrix(Sig2s, 6, 3, Sig2p);
  /*cout<<"Les matrices sigma' etendues sont bien
definie."<<endl;
  cout<<"Voici Sig0s :"<<endl<<Sig0s<<endl;*/
```

```
  /*on procede au chgt de variable en theta3*/
  STDLAP A(12,12), B(12,12), C(12,12);
  C = -Sig0s-Sig1s;
  B = -2*Sig2s;
  A = -Sig1s;

  /*on se ramene a un probleme de vecteurs propres generalise*/
  STDLAP ZERO(12,12), Ident(12,12);
  Init_STDLAP(ZERO);
  SetId(Ident);
  STDLAP M1(24,24), M2(24,24);
  SetSubMatrix(M1, 0, 0, Ident);
  SetSubMatrix(M1, 0 ,12, ZERO);
  SetSubMatrix(M1, 12, 0, ZERO);
  SetSubMatrix(M1, 12, 0, A);
  SetSubMatrix(M2, 0, 0, ZERO);
  SetSubMatrix(M2, 0 ,12, Ident);
  SetSubMatrix(M2, 12, 0, C);
  SetSubMatrix(M2, 12, 0, B);
  cout<<"Affectation des matrices du pb de vect propres gene
effectuee"<<endl;
  STDLAP REIG; TAB VV(24);
  REIG = Eigenvct2(M1,M2, Real(),VV);
  cout<<REIG<<endl<<VV<<endl;

}
```

.3.2 Fichier de déclaration des types et d'affectation des matrices

```
#include "_Matrix.H"
#include "_Lapack.H"


#define TAB VectStd<array1d<double> >
#define MAT MatStd<array2d<double> >
#define STDLAP MatStd<lapack<double> >
#define LAP lapack<double>

/**R sera le tableau retourne par la procedure, les autres parametres
de la fonction sont des donnes de l'utilisateur. C'est la matrice Q de
l'article**/
```

```
void  affectmatun(LAP &R , TAB lambda, TAB mu, TAB a, TAB d, MAT MM)

void affectmatdeux(LAP &O,TAB lambda,TAB mu,TAB a,TAB d)

void affectmattrois(LAP &O,TAB lambda,TAB mu,TAB a,TAB d)

void affectmatquat(LAP &O,TAB lambda,TAB mu,TAB a,TAB d)
```

.3.3 Déclarations de fonctions spécifiques : mymat.H

```
#ifndef _mymat_H_
#define _mymat_H_
#include "_Lapack.H"

// fonction qui initialise une matrice remplie de zero
void Init_STDLAP(STDLAP &A)
{
  unsigned int i,j;
  for(i = 0; i < A.nbrow(); i++)
    {
      for(j = 0; j < A.nbcol(); j++)
        {
          A(i,j) = 0;
        }
    }
}


//fonction qui remplie une matrice carre avec l'identite
void SetId(STDLAP &A)
{
  unsigned int i,j;
  if(A.nbrow() == A.nbcol()){
    for(i = 0; i < A.nbrow(); i++){
      for(j = 0; j < A.nbcol(); j++){
        if(i == j){
          A(i,j) = 1;
        }
        else{
          A(i,j) = 0;
        }
      }
    }
  }
```

```
  else{
    cout<<"Error in SetId, argument must be a square
 matrix"<<endl;
  }
}


// Cette fonction sert a definir des matrices bloc par bloc

void SetSubMatrix(STDLAP &A, int a, int b, STDLAP B)
{
  unsigned int i,j;
  for(i = 0; i < B.nbrow(); i++)
  {
    for(j = 0; j < B.nbcol(); j++)
    {
      A(a + i, b + j) = B(i,j);
    }
  }
}

 void reigvct2(lapack<double> & E, lapack<double> & A, lapack<double>
& B, const Real &);

template <class M> inline
MatRef<lapack<typename M::coef_type> >
Eigenvct2(M & A, M & B, Real) {
  typedef lapack<typename M::coef_type> E;
  E* x = new E(A.nbcol(),A.nbrow());
  reigvct2(*x, A.rep, B.rep, Real());
  return MatRef<E>(x);
}

template <class M,class V> inline
MatRef<lapack<typename M::coef_type> >
Eigenvct2(M & A, M & B, Real, V & v) {
  typedef lapack<typename M::coef_type> E;
  E* x = new E(A.nbcol(),A.nbrow());
  reigvct2(v.rep,*x, A.rep, B.rep, Real());
  return MatRef<E>(x);
}


#include "mymat.C"
```

```
#endif _mymat_H_
```

.3.4 Le fichier mymat.C

```
/*fonction qui rend 1 si l'argument est une des coordonnees du
 vecteur et 0 sinon */
int appartient(double nb, TAB V)
{
  int res = 0;
  int i = 0;
  int taille = V.size();
  while((i < taille) && (res == 0)){
    if(V[i] == nb) res =1;
    i++;
  }
  return(res);
}


/*fonction qui extrait un sous-vecteur de V de l'indice a a l'indice b*/
TAB sub_vect(TAB &V, int a,int b)
{
  TAB RES(b - a);
  for(int i = 0; i < (b - a); i++){
    RES[i] = V[a + i];
  }
  return(RES);
}


/*fonction qui retourne une sous matrice de l'indice (a,b) a celui (c,d)*/
STDLAP Sub_Matrix(STDLAP A, int a,int b,int c,int d)
{
  STDLAP RES(c - a, d - b);
  for(int i = 0; i < c - a; i++){
    for(int j = 0; j < d - b; j++){
      RES(i , j)= A(i + a, j + b);
    }
  }
  return(RES);
}
```

```
/*fonction qui compte le nombre de coordonnees distincts*/
int number_distincts_coord(TAB V)
{
  int taille,res,in;
  taille = V.size();
  TAB tmp(taille - 1);
  res = 0;
  if(taille > 1){
      tmp = sub_vect(V, 1, taille);
      in = appartient(V[0], tmp);
      if(in == 0){
        res = 1 + number_distincts_coord(tmp);
      }
      else{
        res = number_distincts_coord(tmp);
      }
  }
  else{
    if(taille == 1) res = 1;
    else res = 0;
  }
  return(res);
}

/*fonction qui retourne un vecteur sans doublon */
TAB No_Double(TAB V)
{
  int taille,j,cond;
  j = 0;
  taille = number_distincts_coord(V);
  int TAILLE = V.size();
  TAB loc,RES(taille);
  for(int i = 0; i < TAILLE; i++){
    loc = sub_vect(V,i + 1,TAILLE);
    cond = appartient(V[i],loc);
    if(cond == 0){
      RES[j] = V[i];
      j++;
    }
  }
  return(RES);
}
```

```
/*fonction de concatenation de matrices*/
/*Attention cette fonction duplique les matrices*/
STDLAP Concat_Matrix(STDLAP A, STDLAP B)
{
  int rowa = A.nbrow();
  int cola = A.nbcol();
  int colb = B.nbcol();
  STDLAP RES(rowa, cola + colb);
  for(int i = 0; i < rowa; i++){
    for(int j = 0; j < cola; j++){
      RES(i,j) = A(i,j);
    }
    for(int j = 0; j < colb; j++){
      RES(i,j + cola) = B(i,j);
    }
  }
  return(RES);
}


/*Calcul du noyaux d'une matrice STDLAP a partir d'une Svd*/
STDLAP Kernel(STDLAP A)
{
  int n_c = A.nbcol();
  int n_r = A.nbrow();
  /*Calcul du rang pour savoir combient de vecteurs singulier il y a*/
  int r = Rank(A);
  STDLAP V(n_r,n_r);
  STDLAP W(n_c,n_c);
  Svd(A,V,W);
  STDLAP RES;
  cout<<"ok"<<endl;
  RES = Sub_Matrix(W,0,r,n_c,n_c);
  return(RES);
}


void eigbuild(array1d<double> & V, lapack<double > & A,
                  int n, double* t, double * beta, double* vr,
```

```
double* vi, Real)
{
  int c=0;
  for(int j = 0; j < n; j++){
    if(vi[j]==0){
      for(int i = 0; i < n; i++) A(i,c)= t[i+n*j];
      V[c] = vr[j]/beta[j];
      c++;
    }
  }
}

void reigvct2(array1d<double> & V, lapack<double> & E, lapack<double>
& A, lapack<double> & B, const Real &)
{
  //   if(A.nbrow_ != A.nbcol_)
  //     Error("%s %s lapack<double>::leigvct MatrTbXX not square !!!
  //\n",ERR_DIM, ERR_FUNCTION);

  integer n = A.nbrow_, lda = A.lda_, ldb = B.lda_;
  integer* info = new integer;
  double*  WR = new double[n];
  double*  WI = new double[n];
  double*  BETA = new double[n];

  double*  VL= new double[A.nbcol_*n];
  double*  VR= new double[A.nbcol_*n];
  integer  lwork = n*n + 6*n;
  double* work  = new double[lwork];
  // Copy of A and B
  double* At = new double[n*n];
  double* Bt = new double[n*n];

  for(int i=0;i<n*n;i++) {
    At[i] = A.tab_[i]; Bt[i] = B.tab_[i];
  }

  F77NAME(dgegv) ("N", "V", &n, At,  &lda, Bt, &ldb, WR, WI, BETA,
                  VL, &n, VR, &n, work, &lwork, info);

  E.rebuild(n,Lapack_count(WI,n));
  eigbuild(V,E,n,VR,BETA,WR,WI,Real());
}
```