Reconstruction de courbes et surfaces à partir de données tangentielles

Nathalie Sprynski

UNIVERSITE JOSEPH FOURIER - GRENOBLE I

Nº attribué par la bibliothèque

THÈSE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITE JOSEPH FOURIER

Spécialité : "Mathématiques et Informatique"

dans le cadre de **l'École Doctorale**
"Mathématiques, Sciences et Technologies de l'Information, Informatique"

présentée et soutenue publiquement par

NATHALIE SPRYNSKI

le 5 Juillet 2007

RECONSTRUCTION DE COURBES ET SURFACES À PARTIR DE DONNÉES TANGENTIELLES

Directeur de thèse :

Bernard LACOLLE

JURY

M. Bernard ESPIAU	Directeur de recherche - INRIA	Président
M. Marc DANIEL	Professeur - Université de la Méditerranée	Rapporteur
M. Marc NEVEU	Professeur - Université de Bourgogne	Rapporteur
Mme. Stefanie HAHMANN	Professeur - INP de Grenoble	Examinatrice
M. Bernard LACOLLE	Professeur - Université Joseph Fourier	Examinateur
M. Dominique DAVID	Habilité à diriger des recherches - CEA/LETI	Examinateur
M. Luc BIARD	Maître de Conférence - Université Joseph Fourier	Examinateur

Thèse préparée au Laboratoire Fonctionnalisation des Matériaux par des Micro et Nano Systèmes (LETI/DCIS/SMOC/LFM2N) et au Laboratoire Jean Kuntzmann (LJK - IMAG)

Résumé

Reconstruction de courbes et surfaces à partir de données tangentielles

Le LETI développe des micro-capteurs (tels que des micro-accéléromètres ou des micro-magnétomètres) capables de se géoréférencer et de donner alors leur propre orientation. Ainsi, si nous posons un ensemble de capteurs sur une forme, ils nous fourniront les données tangentielles de la surface aux positions des capteurs. Le travail développé ici consiste à reconstruire la forme d'objets à partir de ces données tangentielles et de la connaissance de la répartition de ces capteurs. Nous étudierons plusieurs cas de reconstruction, en commençant par la reconstruction de courbes planes, puis de courbes gauches et enfin de surfaces. Puis nous proposerons des méthodes de capture de mouvement de formes en déformation, c'est-à-dire que nous allons équiper une courbe ou une surface de capteurs, nous allons lui appliquer des déformations, et nous devrons reconstruire la forme virtuelle associée en temps réel à partir des données fournies par les capteurs. De nombreuses applications sont envisagées (domaines de la santé, de l'aéronautique, du multimedia, ...), et nous élaborons en parallèle des prototypes de capture de forme afin de tester et valider nos méthodes.

Mots-clés : capture de formes, capture de mouvement, réseau de capteurs, données tangentielles, capteurs d'orientation, reconstruction de courbes et surfaces, interpolation, paramétrisation curviligne, géodésiques

---

Abstract

Curve and surface reconstruction via tangential information

Micro-sensors developed at the LETI (like microaccelerometers or micromagnetometers) are able to give some information about their orientation. So if we put an array of sensors on a object, they will give data about the local tangency of the object. This work consists in reconstructing the shape of the object thanks to these sensors informations. The shape can be a curve lying in a plane, or a space curve, and then a surface. Then we propose the motion capture of a shape in deformation, i.e. we will equip a curve or a surface with sensors, make movements and deformations with it, and reconstruct it in the same time via data from sensors. There is a lot of applications (medical, aeronautic, multimedia, hobbyist - do -it - yourself applications), and some materials will be experimented in the same time to test and validate these algorithms.

Key-words : motion capture, shape capture, mesh of sensors, tangential informations, sensors of orientation, curve and surface reconstruction, interpolation, arc-length parameterization, geodesics

Remerciements

Je tiens tout d'abord à remercier les personnes qui ont rendu possible cette thèse : Dominique David, *qui a eu l'idée de venir voir du coté des mathématiques appliquées pour répondre à ces questions de capture de forme, et Bernard* Lacolle *et Luc* Biard *pour avoir répondu présent et avoir cru à ce projet. Ils m'ont ensuite accompagnés durant ces trois années avec beaucoup de disponibilité et d'enthousiasme !*

Je tiens également à remercier Marc Daniel *et Marc* Neveu *pour avoir acceptés de rapporter ce travail, ainsi que les autres membres du jury Stéfanie* Hahmann *et Bernard* Espiau.

Il me serait maintenant impossible de noter ici toutes les personnes que j'aimerais remercier pour avoir été présent : Roland Blanpain pour m'avoir accueillie dans son service, Dominique Vicard pour avoir voulu que cela continue, Virginie, Stéphanie et Caroline pour m'avoir guidée dans les méandres administratifs du CEA, et plus généralement toutes les personnes que j'ai cotoyées au sein du LETI, avec lesquelles j'ai travaillé, ou simplement partagé du temps et qui ont rendu ces trois années très agéables et enrichissantes, avec une pensée particulière pour Jean-Charles, Patrick, Laurent, Marina, Emilie, Cédric, Lolo et tous les autres...

Enfin, un grand merci à mes parents et mes frère et soeurs qui ont tout fait pour que j'en sois là aujourd'hui, et je ne pourrais finir sans parler de Pascal et Maëlle : merci d'être là...

Table des matières

INTRODUCTION

Ce travail de thèse en Mathématiques et Informatique a été effectué dans le service des Microsystèmes et Objets Communicants (SMOC) au CEA-LETI (*Commissariat à l'Energie Atomique - Laboratoire d'Electronique et Technologies de l'Information*), centre reconnu de recherche technologique de Grenoble. Ces travaux sont donc le résultat d'une collaboration entre des personnes issues de deux mondes différents : d'un côté des ingénieurs chercheurs du CEA qui cherchent une solution concrète à des problèmes posés par des applications technologiques, et de l'autre côté, des chercheurs en mathématiques appliquées à l'université qui mettent en place des solutions théoriques à des problèmes mathématiques. Cette alliance a donc permis un élargissement et une richesse dans les deux mondes : d'une part essayer de trouver d'autres méthodes pour la résolution concrète de problèmes dictés par les applications et, d'autre part, s'ouvrir un nouveau champ de problématique afin d'y expérimenter de nouvelles solutions théoriques.

En ce qui nous concerne, le LETI développe depuis de nombreuses années des capteurs de données terrestres (champ magnétique, champ gravitationnel...). En les combinant, nous obtenons des systèmes de mesures capables de se géoréférencer. Un grand champ applicatif s'ouvre alors : la capture de mouvement. En effet, ces dispositifs de mesures placés sur des objets donnent l'orientation desdits objets et associés à des hypothèses à priori permettent d'obtenir des informations de position relative et de mouvement. Ces systèmes montrent leurs nombreux avantages : assez petits, ils sont non intrusifs et permettent d'être utilisés partout (car ne nécessitent aucun appareillage extérieur de référence, ils utilisent des références terrestres). Par exemple, si nous essayons de reconstruire des mouvements humains, nous pouvons en disposer en des positions stratégiques du corps et utiliser un modèle de squelette pour arriver au but recherché.

La miniaturisation de ces systèmes permet d'ouvrir d'autres perspectives. En effet, nous pouvons maintenant les utiliser avec une densité plus importante et extraire des informations continues de ces formes en mouvement (en essayant de se passer d'hypothèses à priori concernant les objets que nous cherchons à suivre ou à modéliser) et de nombreux champs applicatifs s'ouvrent dans des domaines très variés, notamment dans les domaines suivants :

- *textile* : avec la miniaturisation des capteurs, il sera bientôt possible de tisser des fils instrumentés directement dans les tissus, nous obtiendrons ainsi des tissus capables de fournir leur propre forme (matériaux dits proprioceptifs) : cela pourrait fournir une aide à la création de vêtements, ou alors être un instrument de mesure surfacique,
- *CAO* : un ruban instrumenté pourrait être vu comme un nouvel outil de modélisation aidant à la création de modèles numériques pour les logiciels de CAO, ou dans le futur le tissu intelligent permettra d'obtenir instantanément la forme numérisée, en

le posant simplement sur la forme voulue,

- *santé* : nous pouvons créer des outils permettant la mesure exacte de la morphologie d'une personne, aidant ainsi soit les diagnostics (par exemple si l'on suit exactement la forme de la colonne vertébrale), soit la création sur mesure de matériels de correction (corsets, bas de contention...),
- *aérodynamique* : ces outils peuvent faciliter les études de perturbations aérodynamiques (par exemple avec des fils instrumentés volants derrière les ailes d'avion dans les souffleries permettant de connaître instantanément les formes de ces rubans en temps réel),
- *sport* : nous pouvons aussi aider à l'amélioration des performances des bateaux en permettant la connaissance en temps réel de la forme de leur voile...

C'est maintenant que l'analyse mathématique et la géométrie différentielle interviennent. La problématique de ce sujet de thèse est donc la suivante : comment reconstruire des formes à partir d'un réseau maillé de capteurs.

Notre travail s'est articulé autour de deux phases distinctes ; la première étant de bien définir le problème, c'est-à-dire : de quoi avons-nous besoin pour la résolution de ce problème, cela est-il possible technologiquement, et enfin a-t-on une bonne modélisation mathématique du problème technologique ? La deuxième phase est une phase plus conventionnelle au niveau mathématique : nous cherchons des solutions à un problème bien posé.

Définissons à présent ce problème de reconstruction de formes à partir de capteurs de façon mathématique. Les capteurs donnent des informations sur leur propre direction, le fait de les poser sur une forme donnera ainsi une information tangentielle à la surface à l'endroit du capteur. Les capteurs ne donnent cependant pas leur position absolue, mais nous aurons comme données des informations relatives à leur répartition (information connue par construction des futurs systèmes de capture : par exemple un ruban instrumenté de capteurs dont nous connaissons la répartition le long de ce fil, ou un filet de capteurs dont nous connaissons les dimensions des mailles...).

Le problème technologique et applicatif de reconstruction de formes à partir de capteurs devient ainsi la formulation mathématique suivante : comment reconstruire une surface à partir de données tangentielles dont la répartition est connue.

Ce problème est bien sûr à définir plus précisément, ce que nous ferons dans la suite de ce document.

Nous articulons ce mémoire de la manière suivante : après un bref chapitre constitué de rappels de notions de géométrie différentielle nécessaires à la compréhension de ce travail, une première partie sera consacrée à la reconstruction de courbes. Cela permettra ainsi de bien poser et comprendre un problème simplifié, et mettre en place des éléments importants qui seront par la suite réinvestis pour la reconstruction des surfaces. Cette partie se décompose en deux chapitres qui forment deux méthodes différentes de résolution dictées par deux visions différentes : en premier lieu, la reconstruction de courbes statiques (dans lequel nous décomposons encore en résolvant tout d'abord le problème des courbes planes, puis nous étendons à la reconstruction des courbes gauches), et dans un second temps nous étudierons la mise en mouvement de notre courbe, qui permettra de disposer d'informations supplémentaires, à savoir la position de la courbe au temps précédent.

Dans une deuxième partie, nous nous pencherons sur la reconstruction de surfaces, toujours en ayant à l'esprit les deux visions : aspects statique ou suivi de déformation. Plusieurs méthodes, dues à différentes possibilités de prototypes dictant différentes sortes de données, sont étudiées.

Enfin, le dernier chapitre se focalisera sur les applications concrètes liées à ces travaux, tout d'abord avec quelques explications sur les capteurs et les prototypes, puis nous montrerons les résultats obtenus avec ces systèmes de capture. Nous avons choisi de placer toutes les applications technologiques dans un dernier chapitre, car nos travaux de thèse étant dans une spécialité " Mathématiques Informatique ", nous voulions avant tout dégager les problèmes sous une forme mathématique, y apporter une réponse et la confronter ensuite aux expérimentations. Cependant, il peut être utile au lecteur de s'y reporter dans un premier temps, afin de mieux appréhender la motivation et le cadre de ce travail. En outre, nous avons également voulu séparer les expérimentations applicatives actuelles car nous voulions apporter des solutions les plus générales possibles (ne dépendant alors pas des prototypes/matériaux actuellement élaborés), ceci expliquant enfin le caractère illustratif de ce dernier chapitre, et légitimant complètement son positionnement.

Notions préliminaires

Dans ce chapitre préliminaire, nous introduisons quelques notions élémentaires de géométrie concernant la description des courbes et des surfaces, et nous décrivons quelques outils de géométrie différentielle, utile pour la compréhension des principaux résultats de ce mémoire.

Courbes

Une courbe de l'espace peut être décrite de plusieurs façons (implicite, explicite, paramétrique notamment). Nous décrivons par la suite le mode que nous utilisons dans notre travail : la description paramétrique des courbes.

Soit C une courbe de l'espace. Il existe alors un paramètre t défini dans un intervalle $[a, b]$, telle que la courbe C soit décrite par la donnée de la fonction f de la façon suivante :

$$\begin{array}{rcl} f : [a,b] & \longrightarrow & \mathbb{R}^3 \\ t & \longmapsto & f(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix}. \end{array}$$

La courbe C est ainsi l'image par la fonction f d'un compact de $\mathbb{R}$.

Si nous associons la fonction suivante σ à la courbe C :

$$\begin{array}{rcl} \sigma : I & \longrightarrow & [0, L], L \text{ longueur de la courbe} \\ t & \longmapsto & \sigma(t) = \displaystyle\int_0^t \left\| f'(u) \right\| du. \end{array}$$

Alors le paramètre $s = \sigma(t)$ est appelé abscisse curviligne et la courbe paramétrée par cette abscisse curviligne (appelée paramétrisation normale) $\overline{f} = f \circ \sigma^{-1}$ est décrite à vitesse constante unitaire, c'est-à-dire :

$$\left\| \overline{f}'(s) \right\| = 1, \ \forall s.$$

Voyons à présent quelques résultats pour les courbes planes.

Le vecteur $\overline{f}'(s)$ est donc unitaire, il peut s'écrire comme fonction d'un angle $\alpha(s)$:

$$\overline{f}'(s) = \begin{pmatrix} x'(s) = \cos(\alpha(s)) \\ y'(s) = \sin(\alpha(s)) \end{pmatrix}$$

Nous pouvons associer le repère de Serret-Frenet $(T(s), N(s))$ en chaque point de la courbe. Il est défini de la façon suivante pour les paramétrisations normales :

$$\begin{aligned} T(s) &= \overline{f}'(s) \text{ unitaire,} \\ N(s) &= e^{i\frac{\pi}{2}} T(s) \text{ afin d'avoir un repère direct.} \end{aligned}$$

A partir de ce repère nous pouvons définir la courbure en chaque point de la courbe (qui est intrinsèque à la courbe) :

$$\kappa(s) = \left\langle \overline{f}''(s), N(s) \right\rangle = \pm \left\| \overline{f}''(s) \right\|$$

Exprimons maintenant la courbure en fonction de l'angle α. Nous devons avoir pour cela les expressions de $N(s)$ et de $\overline{f}''(s)$:

$$\begin{aligned} \overline{f}''(s) &= \begin{pmatrix} -\sin(\alpha(s))\alpha'(s) \\ \cos(\alpha(s))\alpha'(s) \end{pmatrix} \\ N(s) &= e^{i\frac{\pi}{2}} T(s) = \begin{pmatrix} -\sin(\alpha(s)) \\ \cos(\alpha(s)) \end{pmatrix} \end{aligned}$$

D'où :

$$\begin{aligned} \kappa(s) &= \left\langle \overline{f}''(s), N(s) \right\rangle \\ &= \left\langle \begin{pmatrix} -\sin(\alpha(s)).\alpha'(s) \\ \cos(\alpha(s)).\alpha'(s) \end{pmatrix}, \begin{pmatrix} -\sin(\alpha(s)) \\ \cos(\alpha(s)) \end{pmatrix} \right\rangle \\ &= \alpha'(s) \end{aligned}$$

La courbure de la courbe est alors exactement la dérivée de l'angle α.

Voyons à présent ces définitions pour les courbes gauches.

Pour les courbes gauches, nous pouvons définir le trièdre de Serret-Frenet $(T(s), N(s), B(s))$ en chaque point de la courbe. En se plaçant en paramétrisation curviligne, nous avons les définitions suivantes :

$$\begin{aligned} T(s) &= \overline{f}'(s) \text{ vecteur tangent unitaire,} \\ N(s) &= \frac{\overline{f}''(s)}{\left\| \overline{f}''(s) \right\|} \text{ vecteur normal unitaire,} \\ B(s) &= T(s) \wedge N(s) \text{ vecteur binormal unitaire.} \end{aligned}$$

Nous appelons plan osculateur le plan défini par (T, N), le plan normal celui porté par les vecteurs (N, B) et le plan rectifiant celui caractérisé par (T, B).

Nous pouvons alors définir deux notions intrinsèques à la courbe, la courbure κ et la torsion τ :

$$\begin{aligned} \kappa(s) &= \left\| \overline{f}''(s) \right\|, \kappa(s) \geq 0 \\ \tau(s) &= -\left\langle \frac{\partial B}{\partial s}(s), N(s) \right\rangle \end{aligned}$$

Nous pouvons aussi déterminer les valeurs des dérivées du trièdre de Serret-Frenet (formules de Serret-Frenet) :

$$\frac{\partial}{\partial s}\begin{pmatrix} T(s) \\ N(s) \\ B(s) \end{pmatrix} = \begin{pmatrix} 0 & \kappa(s) & 0 \\ -\kappa(s) & 0 & \tau(s) \\ 0 & -\tau(s) & 0 \end{pmatrix} . \begin{pmatrix} T(s) \\ N(s) \\ B(s) \end{pmatrix}$$

Nous allons à présent poser les bases de la description des surfaces.

SURFACES

Une surface peut également être déterminée de façon paramétrique avec cette fois-ci un espace de paramétrisation dans $\mathbb{R}^2$ (nous allons nous limiter à des espaces rectangulaires) :

$$\begin{aligned} F : [a,b] \times [c,d] &\longrightarrow \mathbb{R}^3 \\ (u,v) &\longmapsto F(u,v) = \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix} . \end{aligned}$$

Les courbes isoparamétriques sont des courbes tracées sur la surface où l'un des deux paramètres est constant, par exemple :

$$C_u^{u_0}(v) = F(u_0, v)$$

est la courbe isoparamétrique dépendant de v pour la valeur u fixée à $u = u_0$.

Définissons à présent une catégorie de courbes particulières sur les surfaces : les géodésiques. Avant cela, nous devons définir le repère de Darboux.

Soit une courbe C décrite par la paramétrisation normale $\overline{f}(s)$ sur une surface S paramétrée par F, nous avons alors $\overline{f}(s) = F(u(s), v(s))$. En chaque point de la courbe nous associons le repère de Darboux $(T(s), g(s), n(s))$ défini de la façon suivante :

- $T(s)$ vecteur tangent unitaire de la courbe,
- $n(s)$ normale à la surface,
- $g(s) = n(s) \wedge T(s)$ vecteur normal géodésique.

Ce repère vérifie les relations de Darboux :

$$\frac{\partial}{\partial s}\begin{pmatrix} T(s) \\ g(s) \\ n(s) \end{pmatrix} = \begin{pmatrix} 0 & \kappa_g(s) & \kappa_n(s) \\ -\kappa_g(s) & 0 & -\tau_g(s) \\ -\kappa_n(s) & -\tau_g(s) & 0 \end{pmatrix} . \begin{pmatrix} T(s) \\ g(s) \\ n(s) \end{pmatrix}$$

qui définit $\kappa_n(s)$ la courbure normale, $\kappa_g(s)$ la courbure géodésique, et $\tau_g(s)$ la torsion géodésique.

Une courbe géodésique est alors caractérisée par le fait que la courbure géodésique est identiquement nulle : $\kappa_g(s) \equiv 0$. En conséquence, d'après les formules de Darboux, nous avons $\frac{\partial T}{\partial s}$ colinéaire à n. Et avec les formules de Serret-Frenet, nous avons $\frac{\partial T}{\partial s}$ colinéaire à N. Ainsi nous en déduisons la caractérisation suivante :

Une courbe régulière définie sur une surface est une courbe géodésique si et seulement si son plan osculateur est normal au plan tangent à la surface en tout point de la courbe.

Nous venons de brièvement rappeler les notions de géométrie qui nous sont utiles pour la suite de ce document. Nous pouvons à présent poser le problème sur lequel nous avons travaillé : la reconstruction de courbes à partir de données tangentielles.

Première partie

Reconstruction de courbes

Nous allons dans cette première partie nous intéresser à la reconstruction de courbes à partir de capteurs d'orientation. Nous devons tout d'abord faire le lien entre les systèmes applicatifs possibles et la modélisation mathématique associée. Nous disposons d'un ruban de capteurs régulièrement répartis qui donnent leur propre orientation. Nous pouvons ainsi en extraire la problématique mathématique suivante : nous devons reconstruire une courbe dans l'espace connaissant les données tangentielles en certains points de la courbe dont nous ne connaissons pas la position, mais la répartition le long de la courbe.

En ce sens, ce n'est pas un problème de type Hermite, c'est-à-dire un problème d'interpolation avec contraintes de tangentes. En effet, dans un tel problème, nous devons créer une courbe qui passe par certaines positions connues avec des tangentes connues. En ce qui nous concerne, nous ne connaissons pas les positions, nous connaissons seulement les directions des tangentes et les distances curvilignes entre les points de mesure le long de la courbe.

Ce n'est pas non plus un problème d'enveloppe ([FAR02a]). Dans un problème d'enveloppe, nous devons reconstruire une courbe tangente en certaines droites de positions connues. Dans notre cas, nous connaissons seulement les directions de ces tangentes mais pas leur position dans l'espace, cependant nous connaissons les distances curvilignes entre les points de contact à ces tangentes.

Cette problématique semble donc assez nouvelle. En effet, un tel problème étant issu de données capteurs donne des contraintes assez peu naturelles mathématiquement, et nous apparait alors relativement neuf dans un contexte de modélisation et reconstruction géométrique. Ce mémoire propose des solutions à ce problème novateur.

Voici l'organisation générale de cette première partie. Dans un premier chapitre, nous nous focalisons sur la résolution du problème que nous venons d'expliquer, à savoir la reconstruction de courbes à partir de données tangentielles. Dans un premier temps, nous étudions le problème pour les courbes planes. Nous en expliquons la modélisation qui nous a permis de développer la résolution de notre problème. Puis la consistance de cette méthode est développée au travers de son interprétation physique et de ses invariances. Nous étudions ensuite sa convergence ainsi que le comportement des résultats en faisant intervenir le nombre de capteurs, puis en insérant du bruit dans les données.

Dans un second temps, nous montrons comment nous avons pu étendre cette méthode pour la reconstruction de courbes gauches, et nous démontrons les mêmes caractéristiques de consistance, convergence et robustesse par rapport à l'insertion de bruit.

Dans un deuxième chapitre, nous étudions une des perspectives naturelles de ce travail : le suivi de courbes en mouvement. Dans ce cas, de nouvelles données sont à notre disposition, à savoir la position de la courbe au moment précédent. Nous élaborons donc une nouvelle méthodologie qui tient compte de ces informations supplémentaires, et nous la comparons à la précédente méthode qui peut être appliquée à chaque pas de temps sans connaissances de positions à priori.

Chapitre 1

Reconstruction de courbes

Nous allons dans ce premier chapitre aborder le problème de reconstruction de courbes à partir de données tangentielles. Nous avons déterminé le problème mathématique à partir d'outils technologiques, c'est-à-dire que nous cherchons à modéliser une courbe qui répond à des contraintes de tangentes en certains points de la courbe dont nous ne connaissons pas les positions mais les répartitions curvilignes le long de cette courbe (ces données étant fournies par des capteurs posés sur une courbe-ruban réelle). Ce genre de problématique n'ayant jamais été traitée, nous proposons ici une méthode de reconstruction en traitant ce problème d'un point de vue géométrique. Ce problème sera tout d'abord traité en se restreignant aux courbes planes. Nous élaborons ainsi une méthodologie à partir d'un problème simplifié, dans lequel la solution revient à interpoler la fonction dérivée (caractérisée par une fonction d'angle), et à intégrer ce résultat. Nous étudions ensuite les différentes caractéristiques des solutions obtenues, dans un premier temps du point de vue de la consistance de la méthode (les courbes obtenues ont-elles un sens physique, sont-elles cohérentes vis-à-vis des invariances ?), nous étudions ensuite le caractère convergent de la solution apportée (les courbes solutions convergent-elles vers les courbes réelles si l'on fait tendre le nombre de capteurs vers l'infini ?), puis nous regardons le comportement des solutions lorsque nous perturbons les données de répartition des points de mesure ou les données tangentielles associées.
Dans un second temps, nous élargissons le problème avec la reconstruction des courbes gauches. Nous verrons comment nous avons étendu la méthodologie développée avec les courbes planes, c'est-à-dire où la solution revient toujours à interpoler la fonction dérivée et à intégrer le résultat. Nous adoptons la même démarche de caractérisation des solutions obtenues afin de valider cette méthode.

La méthodologie de reconstruction des courbes sera ensuite la base de notre travail pour la reconstruction des surfaces.

Sommaire

1.1 COURBES PLANES

1.1.1 DÉFINITION DU PROBLÈME

Le problème en deux dimensions est le suivant : nous devons reconstruire une courbe dans le plan en ayant seulement des informations concernant son espace tangentiel. De façon plus précise, nous devons reconstruire une courbe Γ, ne connaissant que les orientations des tangentes en certains points d'échantillonnage (nous connaissons la répartition des capteurs et les angles des tangentes en ces points le long de la courbe à reconstruire).

Supposons que nous ayons $(n+1)$ capteurs le long de la courbe, alors en utilisant le paramétrage par l'abscisse curviligne pour la fonction φ associée à la courbe Γ, nous connaissons seulement les angles $(\alpha_i = \alpha(s_i), i = 0, ..., n)$ pour des valeurs données $(s_i, i = 0, ..., n)$ de l'abscisse curviligne, où $\alpha(s) = (e_1, \varphi'(s))$ est l'angle entre l'axe des x et le vecteur tangent (unitaire) à la courbe (voir FIG.1.1).

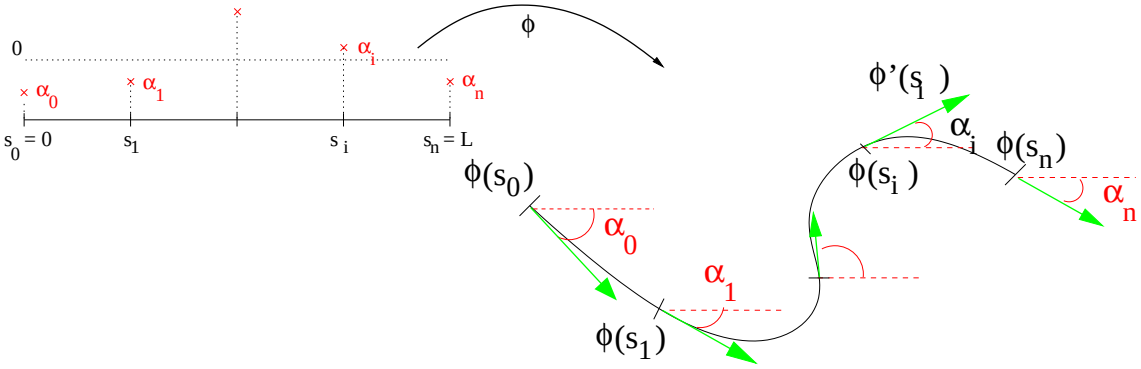


FIG. 1.1 – Définition du problème

1.1.2 MODÉLISATION

La définition du problème nous invite à chercher une solution paramétrée par l'abscisse curviligne. Nous cherchons donc une solution de la forme :

$$C(s) = \begin{pmatrix} x(s) \\ y(s) \end{pmatrix}. \tag{1.1}$$

Si la fonction C est suffisamment dérivable, alors sa dérivée est un vecteur unitaire (propriété de l'abscisse curviligne), elle peut donc se paramétrer en fonction d'un angle $\alpha(s)$:

$$C'(s) = \begin{pmatrix} x'(s) = \cos(\alpha(s)) \\ y'(s) = \sin(\alpha(s)) \end{pmatrix}. \tag{1.2}$$

Les données que nous avons sont la valeur de l'angle en certains points de la courbe initiale dont l'échantillonnage est connu :

$$s_i \to \alpha(s_i) := \alpha_i, \{s_i, i = 0, ..., n\} \text{ connus}$$

De cette modélisation découle immédiatement la méthode de résolution qui se décompose en deux étapes :

- tout d'abord reconstruire la fonction des angles $\alpha(s)$ telle que $\{\alpha(s_i) = \alpha_i, i = 0, ..., n\}$: nous connaissons ainsi la fonction dérivée $C'(s)$,
- puis intégrer la fonction dérivée pour obtenir la courbe solution.

La deuxième partie étant bijective (à partir d'une fonction dérivée et d'un point de départ, il y a unicité de la courbe intégrée), tous les choix de reconstruction se trouvent dans la phase de reconstruction des angles. Nous allons voir par la suite le choix de notre reconstruction de fonction d'angles, ainsi que les conséquences.

A. INTERPOLATION

Posons tout d'abord des contraintes permettant la reconstruction de cette fonction. Les angles sont des données connues à 2π près (en effet, nous connaissons par les capteurs les orientations des vecteurs tangents, que nous traduisons par des données angulaires connues alors dans un intervalle de 2π). Nous avons donc des incertitudes possibles (voir FIG.1.2).

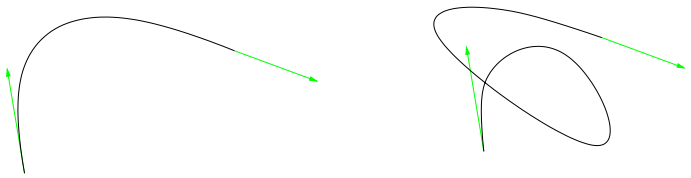


FIG. 1.2 – Deux courbes possibles ayant les mêmes données tangentielles aux extrémités

Nous devons alors effectuer un prétraitement sur les données afin de reconstruire une fonction des angles continue. Pour cela (nous devons bien sûr supposer que notre échantillonnage tient compte du comportement de la fonction des angles), nous allons modifier les valeurs des angles modulo 2π afin de ne plus faire apparaitre de "sauts" : nous allons contraindre deux angles consécutifs à avoir une différence inférieure à π. Mathématiquement, cela va s'écrire de la façon suivante :

pour k de 1 à n :
    si $\alpha_k - \alpha_{k-1} \geq \pi$:
        pour j de k à n : $\alpha_j := \alpha_j - 2\pi$
    si $\alpha_{k-1} - \alpha_k \geq \pi$:
        pour j de k à n : $\alpha_j := \alpha_j + 2\pi$

Ainsi, nous réétalonnons les valeurs des angles. Nous en voyons une illustation dans la figure FIG.1.3.

Ce réétalement des données satisfait à l'hypothèse la plus naturelle si la courbe est assez régulière et si nous avons un nombre de points de mesure significatifs pour la courbe à reconstruire. Ainsi, les données sont bien déterminées, et nous devons donc reconstruire une courbe passant par les points $\{(s_i, \alpha_i), i = 0, ..., n\}$.

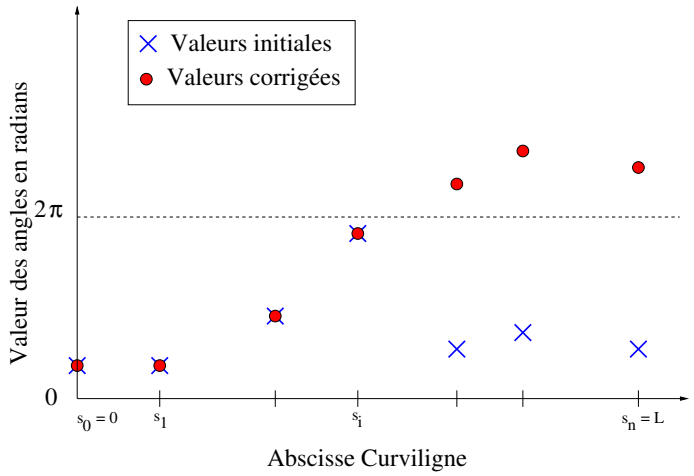


FIG. 1.3 – Modification des angles

Nous avons étudié beaucoup de méthodes de reconstruction, des méthodes d'interpolation par splines polynomiales de degrés divers, des méthodes d'interpolation par contraintes de formes (splines polynomiales ou splines hyperboliques) ainsi que des méthodes d'approximation par splines de lissage. Toutes ces méthodes sont détaillées en annexe A. Ici, nous allons détailler la méthode que nous utilisons car elle est cohérente au sens physique, et satisfaisante dans les exemples réalisés par les démonstrateurs. Cette méthode est l'interpolation de la courbe des angles par une spline cubique naturelle ([BOO78]).

Nous souhaitons interpoler la fonction des angles avec une spline de degré 3 par morceaux, elle s'écrit alors de la façon suivante :

$$\alpha(s) = a_i \left(\frac{s - s_i}{h_i}\right)^3 + b_i \left(\frac{s - s_i}{h_i}\right)^2 + c_i \left(\frac{s - s_i}{h_i}\right) + d_i \tag{1.3}$$

pour s, tel que $s_i \leq s \leq s_{i+1}, i = 0, ..., n-1$ et $h_i = s_{i+1} - s_i, i = 0, ..., n-1$.

La spline cubique que nous cherchons a les contraintes suivantes (appelée spline cubique naturelle) :

- interpolation aux points :

$$\alpha(s_i^+) = \alpha_i, i = 0, \ldots, n-1 \text{ et } \alpha(s_i^-) = \alpha_i, i = 1, \ldots, n,$$

- continuité C^1 :

$$\alpha'(s_i^+) = \alpha'(s_i^-), i = 1, ..., n-1,$$

- continuité C^2 :

$$\alpha''(s_i^+) = \alpha''(s_i^-), i = 1, ..., n-1,$$

- minimisation d'énergie :

$$\alpha''(0) = 0 \text{ et } \alpha''(L) = 0.$$

Les différentes contraintes donnent les équations suivantes :

$$\begin{aligned}\alpha(s_i^+) &= d_i = \alpha_i, i = 0, \ldots, n-1,\\ \alpha(s_i^-) &= a_{i-1} + b_{i-1} + c_{i-1} + d_{i-1} = \alpha_i, i = 1, \ldots, n.\end{aligned}$$

De plus, $\alpha'(s) = \dfrac{3a_i}{h_i}\left(\dfrac{s-s_i}{h_i}\right)^2 + \dfrac{2b_i}{h_i}\left(\dfrac{s-s_i}{h_i}\right) + \dfrac{c_i}{h_i}$, donc :

$$\alpha'(s_i^-) = \frac{3a_{i-1}}{h_{i-1}} + \frac{2b_{i-1}}{h_{i-1}} + \frac{c_{i-1}}{h_{i-1}} = \frac{c_i}{h_i} = \alpha'(s_i^+),$$

et $\alpha''(s) = \dfrac{6a_i}{h_i^2}\left(\dfrac{s-s_i}{h_i}\right) + \dfrac{2b_i}{h_i^2}$, donc :

$$\alpha''(s_i^-) = \frac{6a_{i-1}}{h_{i-1}^2} + \frac{2b_{i-1}}{h_{i-1}^2} = \frac{2b_i}{h_i^2} = \alpha''(s_i^+).$$

Nous obtenons :

$$\begin{aligned} d_i &= \alpha_i,\ i = 0, \ldots, n-1, && \text{(1.4a)}\\ a_i + b_i + c_i &= \alpha_{i+1} - \alpha_i\ , i = 0, \ldots, n-1, && \text{(1.4b)}\\ 3a_i + 2b_i + c_i &= \left(\frac{h_i}{h_{i+1}}\right) c_{i+1},\ i = 0, \ldots, n-2, && \text{(1.4c)}\\ 6a_i + 2b_i &= \left(\frac{h_i}{h_{i+1}}\right)^2 2b_{i+1},\ i = 0, \ldots, n-2. && \text{(1.4d)}\end{aligned}$$

Définissons les $\lambda_i, i = 0, ..., n$ comme étant les dérivées en s_i, $(\alpha'(s_i) = \lambda_i, i = 0, ..., n)$, nous allons exprimer les a_i ,b_i, c_i, en fonction des λ_i, puis nous aurons un système de taille $n+1$ à résoudre. D'après (1.4b) et (1.4c), nous avons :

$$\begin{aligned} c_i &= \lambda_i\ h_i, i = 0, \ldots, n-1, && \text{(1.5a)}\\ a_i &= (\lambda_{i+1} + \lambda_i)h_i - 2(\alpha_{i+1} - \alpha_i\ ),\ i = 0, \ldots, n-1, && \text{(1.5b)}\\ b_i &= 3(\alpha_{i+1} - \alpha_i\ ) - (\lambda_{i+1} + 2\lambda_i)h_i,\ i = 0, \ldots, n-1. && \text{(1.5c)}\end{aligned}$$

Ainsi, l'équation (1.4d) devient :

$$\lambda_{i+2}\left(\frac{1}{h_{i+1}}\right) + \lambda_{i+1}\left(\frac{2}{h_{i+1}} + \frac{2}{h_i}\right) + \lambda_i\left(\frac{1}{h_i}\right) = 3\left(\frac{\alpha_{i+2} - \alpha_{i+1}}{h_{i+1}^2} + \frac{\alpha_{i+1} - \alpha_i}{h_i^2}\right),\ i = 0, \ldots, n-2 \quad \text{(1.6)}$$

et les contraintes de minimisation d'énergie ajoutent :

$$\begin{aligned} \frac{\lambda_1}{h_0} + 2\frac{\lambda_0}{h_0} &= 3\frac{(\alpha_1 - \alpha_0\ )}{h_0^2}, && \text{(1.7a)}\\ 2\frac{\lambda_{n-1}}{h_{n-1}} + \frac{\lambda_n}{h_{n-1}} &= 3\frac{(\alpha_n - \alpha_{n-1}\ )}{h_{n-1}^2}. && \text{(1.7b)}\end{aligned}$$

Nous obtenons grâce à (1.6) et (1.7) le système matriciel de taille $(n+1)$ suivant : $M\Lambda = B$, avec :

$$M = \begin{pmatrix} 2\frac{1}{h_0} & \frac{1}{h_0} & 0 & \cdots & \cdots & \cdots & 0 \\ \frac{1}{h_0} & 2\frac{1}{h_0}+2\frac{1}{h_1} & \frac{1}{h_1} & 0 & \cdots & \cdots & 0 \\ 0 & \frac{1}{h_1} & 2\frac{1}{h_1}+2\frac{1}{h_2} & \frac{1}{h_2} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & 0 & \frac{1}{h_{n-1}} & 2\frac{1}{h_{n-1}}+2\frac{1}{h_n} & \frac{1}{h_n} \\ 0 & \cdots & \cdots & \cdots & 0 & \frac{1}{h_n} & 2\frac{1}{h_n} \end{pmatrix}, \tag{1.8}$$

$$\Lambda = \begin{pmatrix} \lambda_0 \\ \lambda_1 \\ \vdots \\ \vdots \\ \lambda_{n-1} \\ \lambda_n \end{pmatrix}, B = 3 \begin{pmatrix} \frac{\alpha_1-\alpha_0}{h_0^2} \\ \frac{\alpha_1-\alpha_0}{h_0^2}+\frac{\alpha_2-\alpha_1}{h_1^2} \\ \frac{\alpha_2-\alpha_1}{h_1^2}+\frac{\alpha_3-\alpha_2}{h_2^2} \\ \vdots \\ \frac{\alpha_{n-1}-\alpha_{n-2}}{h_{n-2}^2}+\frac{\alpha_n-\alpha_{n-1}}{h_{n-1}^2} \\ \frac{\alpha_n-\alpha_{n-1}}{h_{n-1}^2} \end{pmatrix}$$

La matrice M est une matrice tridiagonale symétrique positive : nous obtenons une solution unique du système. En reportant les λ_i dans les formules (1.5) et (1.4a), nous obtenons les inconnues qui déterminent la spline. Nous avons ainsi une interpolation unique des angles avec une spline cubique C^2.

Une fois la courbe des angles reconstruite, nous connaissons parfaitement la fonction dérivée de la courbe que nous cherchons :

$$C'(s) = \begin{pmatrix} x'(s) = \cos(\alpha(s)) \\ y'(s) = \sin(\alpha(s)) \end{pmatrix}.$$

Il nous reste à présent à intégrer ces deux composantes pour obtenir la solution.

B. INTÉGRATION

Une fois la fonction $\alpha(s)$ déterminée, nous pouvons trouver la solution unique par intégration :

$$C(s) = V_0 + \begin{pmatrix} \int_0^s \cos(\alpha(t))dt \\ \int_0^s \sin(\alpha(t))dt \end{pmatrix} \tag{1.9}$$

où V_0 est la position initiale. Nous avons donc à calculer $\int_0^s \cos(\alpha(t))dt$ et $\int_0^s \sin(\alpha(t))dt$.

Pour s entre s_i et s_{i+1}, nous avons :

$$\begin{aligned} \int_0^s \cos(\alpha(t))dt &= \int_0^{s_i} \cos(\alpha(t))dt + \int_{s_i}^s \cos(\alpha(t))dt \\ &= \sum_{k=0}^{i-1} \int_{s_k}^{s_{k+1}} \cos(\alpha(t))dt + \int_{s_i}^s \cos(\alpha(t))dt \end{aligned} \tag{1.10}$$

Avec un changement de variables, nous obtenons :

$$\int_{s_i}^{s} \cos(\alpha(t))dt = h_i \int_0^{\frac{s-s_i}{h_i}} \cos\left(\alpha(h_i.u + s_i)\right) du. \tag{1.11}$$

Nous obtenons ainsi une fonction à intégrer entre 0 et $\frac{s-s_i}{h_i}$ qui est inférieur à 1. Nous aurons les calculs similaires pour l'intégrale en sinus. La méthode que nous utilisons pour l'intégration est une méthode numérique : la méthode de Simpson (voir [DEM96]).

La méthode de Simpson est une méthode d'intégration numérique, avec un pas à fixer selon l'erreur maximale tolérée :

$$\int_a^b f(t)dt = \frac{h}{3}\left(f_1 + f_{n+1} + 4\sum_{\substack{i=2 \\ i \text{ pair}}}^{n-1} f_i + 2\sum_{\substack{i=3 \\ i \text{ impair}}}^{n} f_i\right) + Err \tag{1.12}$$

avec

$$\begin{aligned} h &= \frac{b-a}{n}, f_i = f\left(a + \frac{i-1}{h}\right), \\ Err &= O(h^4). \end{aligned}$$

Nous allons devoir déterminer les paramètres à fixer (soit h, soit n).

Lors de l'application de cette méthode dans notre travail, puisque nous réalisons des changements de variables, nous avons à intégrer entre 0 et t, où t est entre 0 et 1. Nous avons donc de grandes variations de la taille de l'intervalle d'intégration, c'est pourquoi nous devons fixer non pas le nombre de pas dans l'intégration (n), mais le pas d'intégration (h) lui-même, nous aurons ainsi un nombre de pas différents selon la taille de l'intervalle.

Nous allons maintenant nous intéresser à la caractérisation de cette méthode, en voyant tout d'abord quelques exemples obtenus grâce à elle, puis nous étudierons ses propriétés.

1.1.3 ERREURS ET QUELQUES COURBES

A. ERREURS CALCULÉES

Afin de caractériser les résultats obtenus, voici les différentes erreurs que nous calculons :

- erreur sur la longueur de la courbe reconstruite :

$$e_L = \left(\frac{|L_{th} - L_{obt}|}{L_{th}}\right) . \ 100 \ \%$$

  où L_{th} est la longueur de la courbe théorique et L_{obt} celle de la courbe obtenue par la méthode de reconstruction,

- erreur sur la position des capteurs :

$$e_{capt} = \left(\frac{1}{L_{th}}\sqrt{\frac{1}{Nb_{capt}}\sum_{k=1}^{Nb_{capt}} \|P_{th}(k) - P_{obt}(k)\|^2}\right) . \ 100 \ \%$$

  où $P_{th}(k)$ et $P_{obt}(k)$ sont les positions théorique et obtenue du capteur k,

– erreur sur la position du point d'arrivée :

$$e_{fin} = \left(\frac{1}{L_{th}} . \| P_{th}(Nb_{capt}) - P_{obt}(Nb_{capt}) \| \right) . \ 100 \ \%$$

– erreur de positionnement global de la courbe (pour N points $(s_k, k = 1, ..., N)$ échantillons sur les 2 courbes de même abscisse curviligne) :

$$e_{pos} = \left(\frac{1}{L_{th}} \sqrt{\frac{1}{N} \sum_{k=1}^{N} \| C_{th}(s_k) - C_{obt}(s_k) \|^2} \right) . \ 100 \ \%$$

où C_{th} désigne la courbe théorique, C_{obt} la courbe obtenue, et $C_{th}(s_k)$ le point de la courbe au paramètre s_k,

– erreur de Hausdorff *normalisée* qui compare également globalement les deux courbes entre elles :

$$e_{Haus} = \frac{1}{L_{th}} \max \left\{ \max_k \left(\min_i \| C_{th}(s_k) - C_{obt}(s_i) \| \right), \max_k \left(\min_i \| C_{obt}(s_k) - C_{th}(s_i) \| \right) \right\} . \ 100 \ \%$$

B. QUELQUES COURBES RÉSULTATS

Voyons à présent quelques résultats graphiques et leurs erreurs associées. Pour chaque exemple, nous tracerons à gauche la courbe à reconstruire et sa reconstruction, et à droite la courbe des angles associée, pour la courbe initiale et pour la courbe reconstruite. Les courbes initiales sont en bleu et les courbes reconstruites sont en rouge.

Dans le test suivant (voir FIG.1.4), nous essayons de reconstruire un cercle avec 4 capteurs (le dernier capteur étant confondu avec le premier).

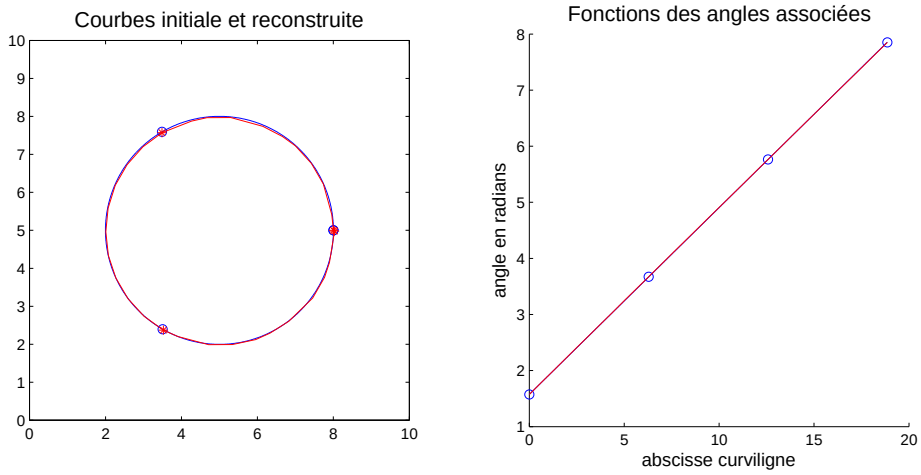


FIG. 1.4 – Reconstruction d'un cercle à partir de 4 capteurs, *(bleu) : courbes initiales, (rouge) : courbes reconstruites*

Les erreurs associées sont les suivantes (voir TAB.1.1) :

e_L	0.34%
e_{capt}	0.14%
e_{fin}	0.18%
e_{pos}	2.25%
e_{Haus}	0.37%

TAB. 1.1 – Erreurs de reconstruction associées au cercle

Nous avons pu remarquer que nous obtenons des erreurs exactement identiques pour tout cercle de rayon quelconque que nous reconstruirions à l'aide de 4 capteurs. Ceci résulte des deux faits suivants : les erreurs ont été normalisées pour être indépendantes de la longueur, et notre méthode est invariante par homothétie (ce que nous verrons par la suite (voir page 38)). Le deuxième exemple montre une spirale avec 30 capteurs équi-répartis (voir FIG.1.5).

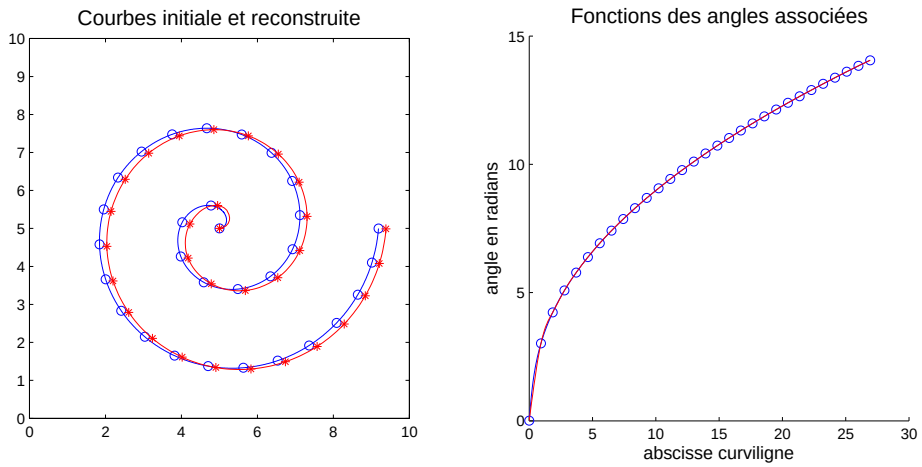


FIG. 1.5 – Reconstruction d'une spirale avec 30 capteurs, *(bleu) : courbes initiales, (rouge) : courbes reconstruites*

Les erreurs associées sont les suivantes (voir TAB.1.2) :

e_L	0.05%
e_{capt}	0.72%
e_{fin}	0.73%
e_{pos}	0.77%
e_{Haus}	0.76%

TAB. 1.2 – Erreurs de reconstruction associées à la spirale

Un troisième exemple montre une spline comportant 12 capteurs (voir FIG.1.6).

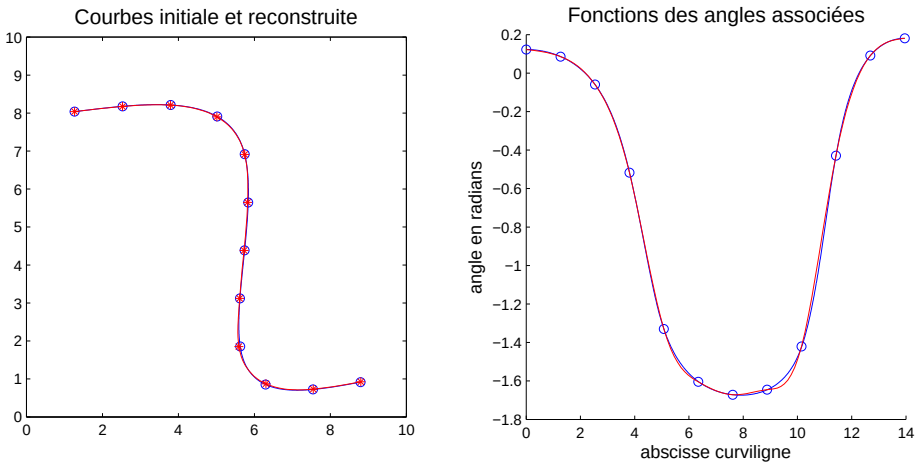


FIG. 1.6 – Reconstruction d'une spline avec 12 capteurs, *(bleu) : courbes initiales, (rouge) : courbes reconstruites*

Les erreurs associées sont les suivantes (voir TAB.1.3) :

e_L	0.02%
e_{capt}	0.11%
e_{fin}	0.09%
e_{pos}	0.60%
e_{Haus}	0.25%

TAB. 1.3 – Erreurs de reconstruction associées à la spline

Enfin un dernier exemple montre une trochoïde avec 42 capteurs (voir FIG.1.7).

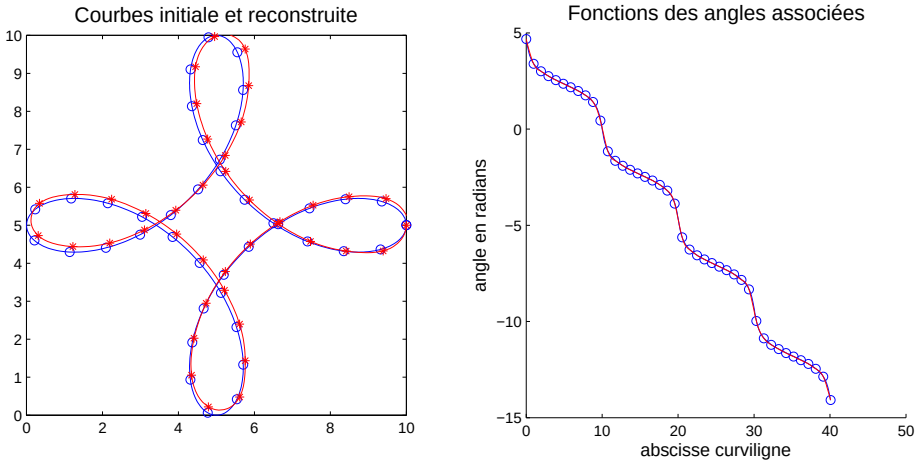


FIG. 1.7 – Reconstruction d'une trochoide avec 42 capteurs, *(bleu) : courbes initiales, (rouge) : courbes reconstruites*

Les erreurs associées sont les suivantes (voir TAB.1.4) :

e_L	0.04%
e_{capt}	0.34%
e_{fin}	0.16%
e_{pos}	0.44%
e_{Haus}	0.56%

TAB. 1.4 – Erreurs de reconstruction associées à la trochoïde

Ces différents exemples montrent que nous obtenons de bonnes reconstructions des courbes, autant du point de vue visuel, qu'avec les erreurs calculées.

Après ces quelques exemples, nous allons voir dans la partie suivante la consistance de cette méthode, ses propriétés puis nous montrerons des résultats de convergence.

1.1.4 CONSISTANCE

A. SENS PHYSIQUE

Voyons ici les influences qu'apportent les contraintes de reconstruction de la courbe des angles par rapport à la reconstruction de la courbe finale.

Nous reconstruisons la courbe des angles par une spline cubique par morceaux, méthode qui minimise l'énergie de flexion de la spline (voir [CAR76]) : nous minimisons

$$\int \alpha''^{\,2}$$

où α représente la fonction angle. Or, dans cette modélisation, puisque nous sommes en paramétrisation curviligne, la courbure de la courbe κ est exactement la dérivee de la fonction angle.

Ainsi, $\kappa = \alpha'$, donc :

$$\min \int \alpha''^{\,2} = \min \int \kappa'^{\,2}. \tag{1.13}$$

Cela signifie que lorsque nous reconstruisons la fonction angle en cherchant à minimiser l'énergie de flexion, nous cherchons à minimiser les variations de courbure de la courbe finale à reconstruire : cette contrainte est très cohérente avec les interprétations physiques des objets que nous cherchons. En effet, nous cherchons à reconstruire des formes issues de matériaux en déformation, donc ayant des propriétés lisses : nous pouvons en conclure que cette méthode est bien adaptée à notre problème.

B. INVARIANCE PAR ROTATION

Cette méthode de reconstruction a également la propriété d'être invariante par rotation. Cela signifie que pour deux courbes initiales identiques à une rotation près, les deux courbes reconstruites par la méthode actuelle seront identiques à la même rotation près. C'est ce que nous allons montrer par la suite.

Soient

$$\left(C^{(1)}(s), \alpha^{(1)}(s)\right) \text{ et } \left(C^{(2)}(s), \alpha^{(2)}(s)\right),$$

deux courbes et leurs fonctions d'angles associées identiques à une rotation près l'une de l'autre. Cela signifie qu'il existe un angle θ tel que

$$\alpha^{(2)}(s) = \alpha^{(1)}(s) + \theta, \forall s.$$

Les données obtenues par le système de capteurs sont une suite de couples (abscisse curviligne, angle) ; les données de la première courbe sont :

$$\{(s_i, \alpha_i^{(1)}), i = 0, ..., n\};$$

celles de la deuxième sont donc :

$$\{(s_i, \alpha_i^{(2)} = \alpha_i^{(1)} + \theta), i = 0, ..., n\}.$$

Nous cherchons à trouver les courbes solutions

$$\left(\widetilde{C^{(1)}}(s), \widetilde{\alpha^{(1)}}(s)\right)$$

reconstruction de $\left(C^{(1)}(s), \alpha^{(1)}(s)\right)$ et

$$\left(\widetilde{C^{(2)}}(s), \widetilde{\alpha^{(2)}}(s)\right)$$

reconstruction de $\left(C^{(2)}(s), \alpha^{(2)}(s)\right)$.

La première étape de la reconstruction de la courbe consiste en l'interpolation de la fonction des angles à partir des échantillons donnés avec une spline cubique. D'après l'équation (1.8), si la fonction $\widetilde{\alpha^{(1)}}(s)$ est solution de l'interpolation des $\{(s_i, \alpha_i^{(1)}), i = 0, ..., n\}$, alors nous obtenons la reconstruction :

$$\widetilde{\alpha^{(2)}}(s) = \widetilde{\alpha^{(1)}}(s) + \theta$$

pour la deuxième courbe. En effet, la recherche de l'interpolant en spline cubique consiste en la recherche d'une suite de quadruplés $\{(a_i, b_i, c_i, d_i), i = 0, ..., n-1\}$. Le calcul des a_i, b_i, c_i utilise seulement les écarts entre angles consécutifs, donc ces valeurs sont identiques pour les 2 courbes, et les valeurs d_i sont les valeurs des angles.

Ainsi la courbe $\widetilde{C^{(2)}}$ reconstruite est de la forme :

$$\widetilde{C^{(2)}}(s) = \begin{pmatrix} \int \cos\left(\widetilde{\alpha^{(2)}}(t)\right) dt \\ \int \sin\left(\widetilde{\alpha^{(2)}}(t)\right) dt \end{pmatrix} = \begin{pmatrix} \int \cos\left(\widetilde{\alpha^{(1)}}(s) + \theta\right) dt \\ \int \sin\left(\widetilde{\alpha^{(1)}}(s) + \theta\right) dt \end{pmatrix} \tag{1.14a}$$

$$= \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \widetilde{C^{(1)}}(s) \tag{1.14b}$$

Nous obtenons bien le résultat suivant : les reconstructions de deux courbes obtenues par rotation l'une de l'autre sont deux courbes identiques à la même rotation près : notre

méthode donne donc des résultats invariants par rotation.

Nous pouvons en voir une illustration dans la figure FIG.1.8, sur laquelle nous voyons sur la gauche les deux courbes initiales en bleu (la deuxième courbe étant obtenue par rotation d'un angle $\frac{\pi}{3}$ de la première courbe) et les deux courbes reconstruites en rouge ; sur le schéma de droite, nous appliquons la rotation inverse sur la deuxième courbe reconstruite afin de la comparer : les deux courbes reconstruites sont confondues (rouge et vert mais la verte seulement est visible), et nous voyons la courbe initiale en bleu.

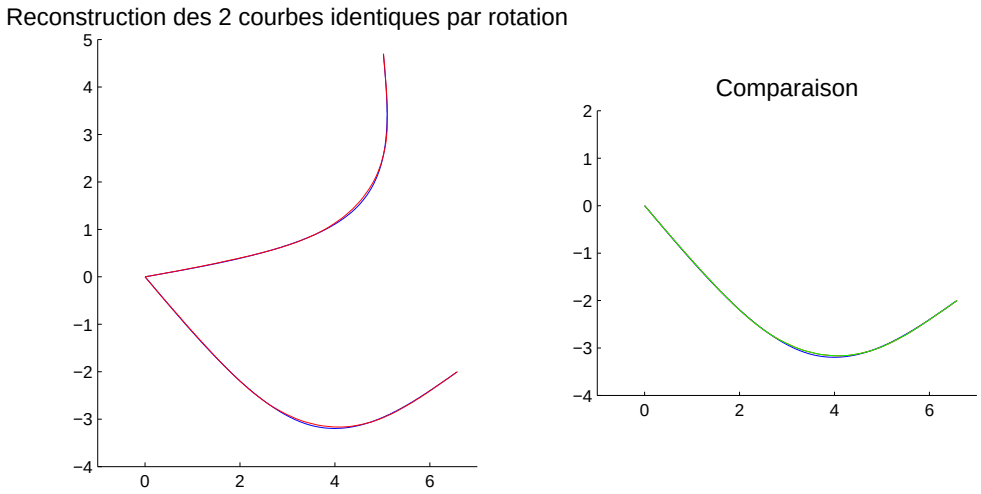


FIG. 1.8 – Invariance par rotation

C. INVARIANCE PAR HOMOTHÉTIE

Un autre propriété importante de cette méthode est son invariance par homothétie, ce qui signifie que pour deux courbes obtenues par changement d'échelle l'une de l'autre, les courbes reconstruites sont identiques à ce même changement d'échelle près. C'est ce que nous allons montrer par la suite.

Soient $C^{(1)}$ et $C^{(2)}$ deux courbes identiques à une homothétie près l'une de l'autre. Cela signifie que si nous les paramétrons par l'abscisse curviligne, il existe un scalaire K tel que nous ayons $C^{(1)}(s)$ définie pour $s \in [0, L]$ et $C^{(2)}(\sigma)$ définie pour $\sigma \in [0, KL]$ avec la relation suivante :

$$C^{(2)}(\sigma) = C^{(2)}(Ks) = K.C^{(1)}(s).$$

Nous illustrons cette caractéristique dans la figure FIG.1.9.

Les données des capteurs sont pour la première courbe :

$$\{(s_i, \alpha_i), i = 0, ..., n\}$$

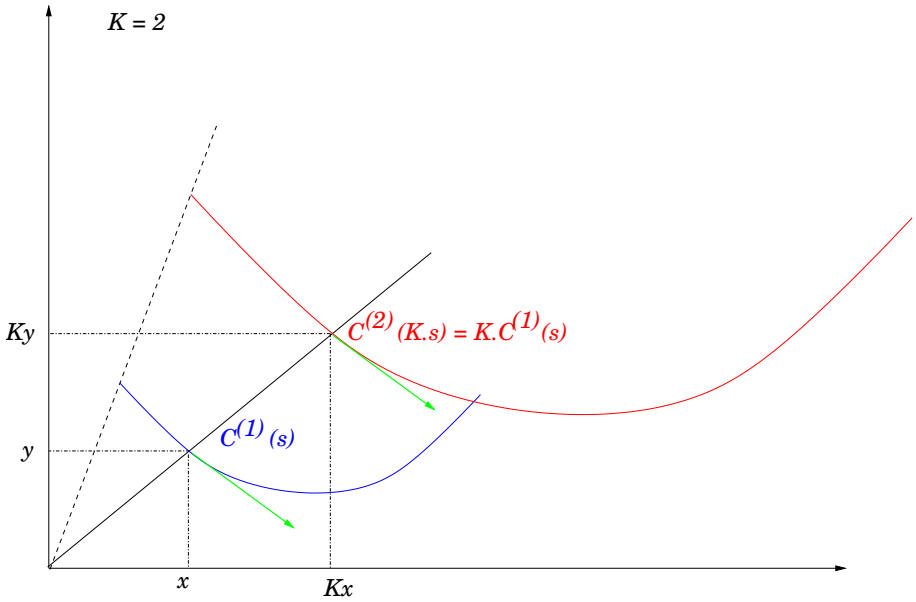


FIG. 1.9 – Illustration de deux courbes homothétiques

et sont donc pour la seconde courbe :

$$\{(\sigma_i = Ks_i, \alpha_i), i = 0, ..., n\},$$

car l'homothétie conserve les tangentes par parallélisme (nous en voyons d'ailleurs l'illustration sur la figure FIG.1.9).

Soient les fonctions $\widetilde{\alpha^{(1)}}(s)$ obtenue pour la courbe 1 et $\widetilde{\alpha^{(2)}}(s)$ obtenue pour la courbe 2, quelle relation y-a-t'il entre elles ? Soit

$$\begin{array}{rcl} \widetilde{\alpha^{(1)}} : [0, L] & \longrightarrow & \mathbb{R} \\ s & \longmapsto & \widetilde{\alpha^{(1)}}(s) \end{array}$$

l'unique solution spline vérifiant les contraintes d'interpolation $\widetilde{\alpha^{(1)}}(s_i) = \alpha_i$. Et considérons ensuite la fonction $\widehat{\alpha^{(2)}}$ définie par :

$$\begin{array}{rcl} \widehat{\alpha^{(2)}} : [0, K.L] & \longrightarrow & \mathbb{R} \\ \sigma & \longmapsto & \widehat{\alpha^{(2)}}(\sigma) := \widetilde{\alpha^{(1)}}\left(\dfrac{\sigma}{K}\right) \end{array}$$

Alors, par construction $\widehat{\alpha^{(2)}}$ est une spline associée aux noeuds $K.s_i$ et elle vérifie les contraintes d'interpolation :

$$\widehat{\alpha^{(2)}}(K.s_i) = \widetilde{\alpha^{(1)}}\left(\frac{K.\sigma}{K}\right) = \alpha_i.$$

Par unicité, cette fonction $\widehat{\alpha^{(2)}}$ est notre interpolation des angles :

$$\widehat{\alpha^{(2)}} = \widetilde{\alpha^{(2)}}.$$

Donc les fonctions angles ont la relation suivante :

$$\widetilde{\alpha^{(2)}}(\sigma) = \widetilde{\alpha^{(1)}}(s) \tag{1.15}$$

Nous pouvons à présent comparer les deux courbes reconstruites :

$$\widetilde{C^{(2)}}(\sigma) = \begin{pmatrix} \int_0^{\sigma} \cos(\widetilde{\alpha^{(2)}}(u)du \\ \int_0^{\sigma} \sin(\widetilde{\alpha^{(2)}}(u)du \end{pmatrix} = \begin{pmatrix} \int_0^{Ks} \cos(\widetilde{\alpha^{(2)}}(u)du \\ \int_0^{Ks} \sin(\widetilde{\alpha^{(2)}}(u)du \end{pmatrix} \tag{1.16a}$$

$$= \begin{pmatrix} \int_0^{s} \cos(\widetilde{\alpha^{(2)}}(Kv)Kdv \\ \int_0^{s} \sin(\widetilde{\alpha^{(2)}}(Kv)Kdv \end{pmatrix} \text{ par changement de variables } v = \frac{u}{K} \tag{1.16b}$$

$$= K. \begin{pmatrix} \int_0^{s} \cos(\widetilde{\alpha^{(1)}}(v)dv \\ \int_0^{s} \sin(\widetilde{\alpha^{(1)}}(v)dv \end{pmatrix} \text{ par la démonstration précédente} \tag{1.16c}$$

$$= K.\widetilde{C^{(1)}}(s) \tag{1.16d}$$

Nous obtenons ainsi deux courbes reconstruites homothétiques l'une de l'autre : la méthode que nous avons mise en place donne donc des solutions invariantes par homothétie. Nous pouvons également en voir une illustration dans la figure FIG.1.10. Nous voyons sur la gauche les deux courbes initiales en bleu et les deux courbes reconstruites en rouge, et la comparaison sur le graphe de droite où les deux courbes reconstruites sont confondues (rouge et vert).

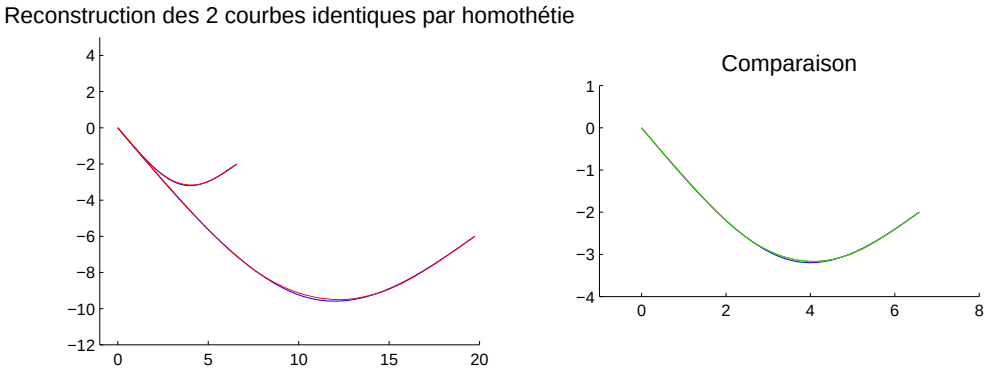


FIG. 1.10 – Invariance par homothétie

1.1.5 CONVERGENCE

Nous allons dans cette partie nous intéresser à la convergence de la méthode. En d'autres termes : la courbe reconstruite va-t'elle converger vers la courbe initiale si nous faisons tendre le nombre de capteurs vers l'infini ? Pour cela, nous procédons en 2 étapes :

- la première étape va consister à majorer les erreurs de reconstruction de la courbe en fonction des erreurs de reconstruction sur la courbe des angles,
- la deuxième étape consiste à déterminer si la méthode de reconstruction de la courbe des angles converge vers la véritable courbe des angles en fonction du nombre de capteurs.

A. CONVERGENCE DE LA COURBE SELON LA CONVERGENCE DE LA COURBE DES ANGLES

Voyons ici quelles hypothèses de convergence nous devons imposer aux angles pour obtenir une convergence de la méthode de reconstruction par intégration. Nous allons majorer différents calculs d'erreur de la courbe à reconstruire en fonction des erreurs de reconstruction de la fonction angle.

Notations : $C = (C_x, C_y)$ la courbe à reconstruire et $\widetilde{C} = (\widetilde{C}_x, \widetilde{C}_y)$ sa reconstruction, $E_\alpha^{(1)}, E_\alpha^{(2)}, E_\alpha^{(\infty)}$ erreurs d'interpolation de la fonction angle en norme L^1, L^2et L^∞.

a. Erreurs sur la courbe à reconstruire

En norme infinie

$$\begin{aligned} \left\|C - \widetilde{C}\right\|_\infty &= \max\left(\left\|C_x - \widetilde{C}_x\right\|_\infty, \left\|C_y - \widetilde{C}_y\right\|_\infty\right) \\ \left\|C_x - \widetilde{C}_x\right\|_\infty &= \sup_{s\in[0,L]} \left|C_x(s) - \widetilde{C}_x(s)\right| \end{aligned}$$

Or :

$$\left|C_x(s) - \widetilde{C}_x(s)\right| = \left|\int_0^s (\cos(\alpha(t)) - \cos(\widetilde{\alpha}(t)))\, dt\right| \leq \int_0^L |\cos(\alpha(t)) - \cos(\widetilde{\alpha}(t))|\, dt$$

De plus,

$$|\cos(a) - \cos(b)| = \left|-2\sin\left(\frac{a+b}{2}\right)\sin\left(\frac{a-b}{2}\right)\right| \leq 2.1.\left|\frac{a-b}{2}\right| \leq |a-b| \qquad (1.17)$$

Donc nous pouvons obtenir 3 majorations différentes (faisant intervenir les 3 normes différentes pour l'erreur de reconstruction de la courbe des angles) :

$$\begin{aligned} \left|C_x(s) - \widetilde{C}_x(s)\right| &\leq \int_0^L |\alpha(t) - \widetilde{\alpha}(t)|\, dt \\ &\leq \sqrt{L}\sqrt{\int_0^L |\alpha(t) - \widetilde{\alpha}(t)|^2 dt} \text{ (inégalité de Cauchy)} \\ \left|C_x(s) - \widetilde{C}_x(s)\right| &\leq \int_0^L \|\alpha(t) - \widetilde{\alpha}(t)\|_\infty\, dt \end{aligned}$$

De même :

$$|\sin(a) - \sin(b)| = \left|2\cos\left(\frac{a+b}{2}\right)\sin\left(\frac{a-b}{2}\right)\right| \leq 2.1.\left|\frac{a-b}{2}\right| \leq |a-b| \tag{1.19}$$

Nous obtenons ainsi le même genre de majoration pour $\left|C_y(s) - \widetilde{C}_y(s)\right|$. Nous obtenons alors une majoration de l'erreur de reconstruction en norme infinie de la courbe en fonction des erreurs de reconstruction de la courbe des angles :

$$\begin{aligned} \left\|C - \widetilde{C}\right\|_\infty &\leq E_\alpha^{(1)} \\ \left\|C - \widetilde{C}\right\|_\infty &\leq \sqrt{L}E_\alpha^{(2)} \\ \left\|C - \widetilde{C}\right\|_\infty &\leq L.E_\alpha^{(\infty)} \end{aligned} \tag{1.20}$$

Ceci nous montre que l'erreur en norme infinie de la courbe à reconstruire est majorée par les erreurs en toute norme de reconstruction de la courbe des angles : si la courbe des angles converge vers la courbe des angles initiale, alors la courbe à reconstruire va converger vers la courbe initiale en norme infinie.

En norme 1

$$\begin{aligned} \left\|C - \widetilde{C}\right\|_1 &= \int_0^L \left\|C(s) - \widetilde{C}(s)\right\|_1 ds \\ \left\|C(s) - \widetilde{C}(s)\right\|_1 &= \left|C_x(s) - \widetilde{C}_x(s)\right| + \left|C_y(s) - \widetilde{C}_y(s)\right| \end{aligned}$$

Et d'après (1.17) et (1.19), nous avons :

$$\begin{aligned} \left\|C - \widetilde{C}\right\|_1 &\leq 2LE_\alpha^{(1)} \\ \left\|C - \widetilde{C}\right\|_1 &\leq 2L\sqrt{L}E_\alpha^{(2)} \\ \left\|C - \widetilde{C}\right\|_1 &\leq 2L^2.E_\alpha^{(\infty)} \end{aligned} \tag{1.21}$$

Nous obtenons le même style de résultats pour l'erreur de reconstruction en norme 1.

En norme 2

$$\begin{aligned} \left\|C - \widetilde{C}\right\|_2 &= \sqrt{\int_0^L \left\|C(s) - \widetilde{C}(s)\right\|_2^2 ds} \\ \left\|C(s) - \widetilde{C}(s)\right\|_2^2 &= \left(C_x(s) - \widetilde{C}_x(s)\right)^2 + \left(C_y(s) - \widetilde{C}_y(s)\right)^2 \end{aligned}$$

Et d'après (1.17) et (1.19), nous avons :

$$\begin{aligned} \left\|C - \widetilde{C}\right\|_2 &\leq \sqrt{2L}E_\alpha^{(1)} \\ \left\|C - \widetilde{C}\right\|_2 &\leq \sqrt{2}LE_\alpha^{(2)} \\ \left\|C - \widetilde{C}\right\|_2 &\leq \sqrt{2L}L.E_\alpha^{(\infty)} \end{aligned} \tag{1.22}$$

Enfin, nous obtenons le même style de résultats pour l'erreur de reconstruction en norme 2.

En conclusion, si la courbe des angles converge (toutes normes confondues), alors la courbe reconstruite converge vers la courbe initiale (toutes normes confondues). Voyons à présent la convergence de la courbe dérivée.

b. Erreurs sur la dérivée de la courbe

$$C'(s) = \begin{pmatrix} \cos(\alpha(s)) \\ \sin(\alpha(s)) \end{pmatrix}$$

En norme infinie

$$\begin{aligned} \left\| C' - \widetilde{C'} \right\|_\infty &= \max\left(\left\| C'_x - \widetilde{C'_x} \right\|_\infty, \left\| C'_y - \widetilde{C'}_y \right\|_\infty \right) \\ \left\| C'_x - \widetilde{C'}_x \right\|_\infty &= \sup_{s\in[0,L]} \left| C'_x(s) - \widetilde{C'}_x(s) \right| \end{aligned}$$

Or :

$$\left| C'_x(s) - \widetilde{C'}_x(s) \right| = |\cos(\alpha(s)) - \cos(\widetilde{\alpha}(s))|$$

Et d'après (1.17) et (1.19), nous avons :

$$\left\| C' - \widetilde{C'} \right\|_\infty \leq E_\alpha^{(\infty)} \tag{1.23}$$

En norme 1

$$\begin{aligned} \left\| C' - \widetilde{C'} \right\|_1 &= \int_0^L \left\| C'(s) - \widetilde{C'}(s) \right\|_1 ds \\ \left\| C'(s) - \widetilde{C'}(s) \right\|_1 &= \left| C'_x(s) - \widetilde{C'}_x(s) \right| + \left| C'_y(s) - \widetilde{C'}_y(s) \right| \end{aligned}$$

Et d'après (1.17) et (1.19), nous obtenons :

$$\begin{aligned} \left\| C' - \widetilde{C'} \right\|_1 &\leq 2E_\alpha^{(1)} \\ \left\| C' - \widetilde{C'} \right\|_1 &\leq 2\sqrt{L}E_\alpha^{(2)} \\ \left\| C' - \widetilde{C'} \right\|_1 &\leq 2L\sqrt{L}.E_\alpha^{(\infty)} \end{aligned} \tag{1.24}$$

En norme 2

$$\begin{aligned} \left\| C' - \widetilde{C'} \right\|_2 &= \sqrt{\int_0^L \left\| C'(s) - \widetilde{C'}(s) \right\|_2^2 ds} \\ \left\| C'(s) - \widetilde{C'}(s) \right\|_2^2 &= \left(C'_x(s) - \widetilde{C'}_x(s) \right)^2 + \left(C'_y(s) - \widetilde{C'}_y(s) \right)^2 \end{aligned}$$

Donc d'après (1.17) et (1.19), nous obtenons :

$$\left\|C'(s) - \widetilde{C'}(s)\right\|_2^2 \leq 2\left|\alpha(s) - \widetilde{\alpha}(s)\right|^2$$

D'où :

$$\begin{aligned} \left\|C' - \widetilde{C'}\right\|_2 &\leq 2.E_\alpha^{(2)} \\ \left\|C' - \widetilde{C'}\right\|_2 &\leq \sqrt{2L}.E_\alpha^{(\infty)} \end{aligned} \tag{1.25}$$

En conclusion, nous avons également convergence de la fonction dérivée en fonction de la fonction des angles : notre méthode de reconstruction de la courbe des angles et son efficacité vont ainsi complètement contrôler la courbe à reconstruire. Voyons donc maintenant si la méthode que nous utilisons pour la reconstruction de la courbe des angles permet une convergence de la courbe obtenue vers la courbe initiale.

B. CONVERGENCE DE LA MÉTHODE D'INTERPOLATION

Nous venons de voir que la convergence de notre méthode dépend de la convergence de la courbe des angles reconstruite vers la courbe des angles réelle de la courbe à reconstruire. Nous allons donc ici nous intéresser à la convergence de la méthode d'interpolation choisie.

Soit la fonction α, définie sur $[0, L]$ au moins C^2, dont nous connaissons les valeurs en $n+1$ points $(\alpha(s_i) = \alpha_i, i = 0, ..., n)$ et que nous devons interpoler. La fonction interpolante $\widetilde{\alpha}$ est une spline cubique naturelle.

Les propriétés de convergence des courbes splines ont longuement été étudiées, et ses résultats apparaissent notamment dans [HAL73],[ATK68], [WEI61] et [JHA63].

Si nous supposons les abscisses s_i régulièrement réparties, alors nous avons le résultat suivant :

$$\max_{s_0 \leq x \leq s_n} |\widetilde{\alpha}(x) - \alpha(x)| = O(h^2), \text{ quand } h \to 0 \tag{1.26}$$

c'est-à-dire que l'interpolation par spline cubique naturelle converge en norme infinie vers la fonction initiale.

C. EN RÉSUMÉ

Dans un pemier temps, nous avons montré le résultat suivant : l'erreur de reconstruction de la courbe finale (que ce soit selon la norme L^1, L^2 et L^∞) est bornée par l'erreur de reconstruction de la fonction des angles (aussi quelle que soit la norme utilisée L^1, L^2 et L^∞).

Dans un deuxième temps, nous venons de voir que l'interpolation par spline cubique naturelle de la fonction des angles convergeait pour la norme infinie vers la solution exacte en fonction du nombre de capteurs.

En résumé, nous venons d'obtenir le résultat suivant : la méthode de reconstruction de courbes à partir de données tangentielles en des points de répartition connue converge vers la solution exacte en fonction du nombre de capteurs.

1.1.6 COURBES D'ERREUR EN FONCTION DE CERTAINS PARAMÈTRES

A. ERREURS EN FONCTION DU NOMBRE DE CAPTEURS

Nous avons vu précédemment que la méthode choisie fait converger les courbes solutions vers les courbes réelles lorsque le nombre de capteurs tend vers l'infini. Ici, nous allons nous intéresser à l'évolution de l'erreur en fonction du nombre de capteurs sur un certain nombre de courbes tests. Les erreurs calculées seront les erreurs que nous avons définies en page 32.

Test 1 Nous testons dans un premier temps un ensemble de courbes représentées sur la figure FIG.1.11.

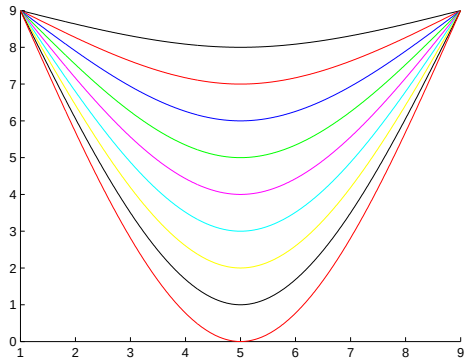
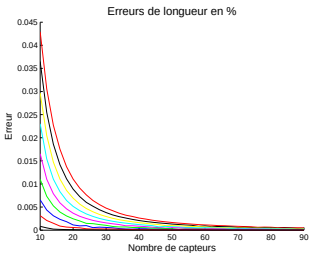
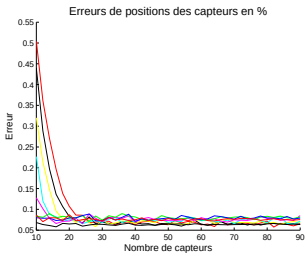


FIG. 1.11 – Ensemble 1 de courbes testées

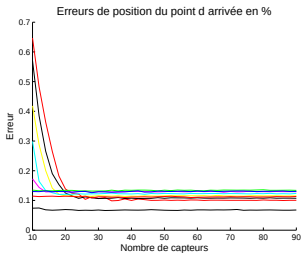
Sur ces courbes, nous avons appliqué notre méthode de reconstruction pour un nombre de capteurs variant de 10 à 90 avec un pas de 2. Voici les différentes courbes d'erreur obtenues (voir FIG.1.12) :



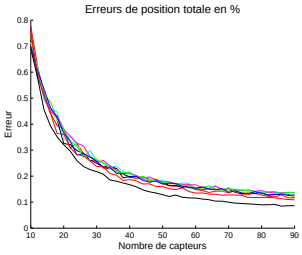
(a) Erreur de longueur



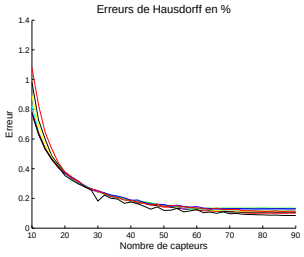
(b) Erreur de position des capteurs



(c) Erreur de position du dernier capteur



(d) Erreur de position totale



(e) Erreur de Hausdorff

FIG. 1.12 – Erreurs pour la spline 1

Ces courbes montrent une décroissance très rapide des erreurs de reconstruction qui permet de voir qu'il n'est pas nécessaire de disposer d'une grande quantité de capteurs pour une bonne courbe solution.

Test 2 Voici le deuxième ensemble de courbes que nous testons (voir FIG.1.13). Nous

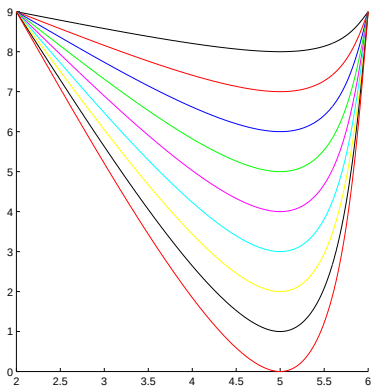
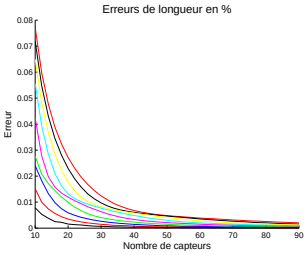


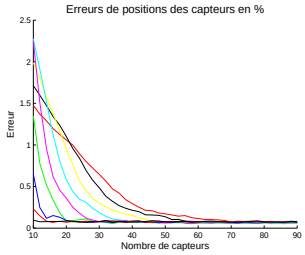
FIG. 1.13 – Ensemble 2 de courbes testées

avons également appliqué notre méthode de reconstruction pour un nombre de capteurs va-

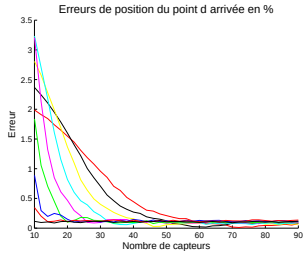
riant de 10 à 90 avec un pas de 2. Les différentes courbes d'erreur obtenues sont rassemblées dans la figure FIG.1.14.



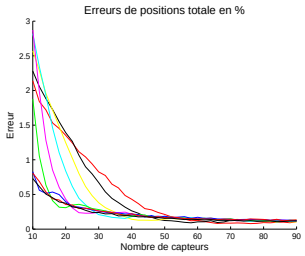
(a) Erreur de longueur



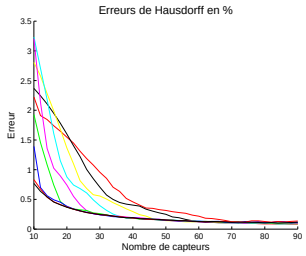
(b) Erreur de position des capteurs



(c) Erreur de position du dernier capteur



(d) Erreur de position totale



(e) Erreur de Hausdorff

FIG. 1.14 – Erreurs pour la spline 2

Nous obtenons la même conclusion avec cet ensemble de courbes. Les erreurs décroissent très vite, la convergence est rapide, nous pouvons utiliser cette méthode de reconstruction pour un nombre de capteurs assez faible.

B. QUELQUES TESTS AVEC DONNÉES BRUITÉES

Nous allons nous intéresser ici aux courbes reconstruites à partir de données bruitées, car les systèmes sur lesquels nous allons utiliser notre algorithme fournira nécessairement des données non parfaites. D'une part, il peut y avoir des erreurs dues au placement des capteurs (positionnement et orientation du capteur sur la forme qui peuvent donc altérer les distances curvilignes et les données des angles), et d'autre part, les capteurs eux-mêmes ont une précision définie (donnée réelle du capteur et traitement pour obtenir l'information tangentielle). Nous avons donc fait des simulations de ces deux types de bruit pour en voir les conséquences sur les courbes obtenues.

a. Bruiter les valeurs des angles Nous allons ici bruiter les angles obtenus par les capteurs par un bruit gaussien de moyenne nulle et de variance variable. Les données seront connues à $\pm k°$, cela signifie que les données seront bruitées avec une variance de $\sigma = k.\frac{\pi}{3.180}$. Voyons ce que nous obtenons pour certaines courbes tests. Chaque point de mesure d'erreur est obtenu en moyennant 10 erreurs sur 10 courbes bruitées de même variance. Nous calculons ici seulement les erreurs de position globale.

Test 1 : le cercle Voyons tout d'abord un résultat de courbe obtenu avec 10 capteurs et des valeurs de capteurs connues à $\pm 10°$ près (voir FIG.1.15) : l'erreur de reconstruction obtenue est de 1.5%. Nous voyons la courbe initiale en bleu, la reconstruction non bruitée en rouge et celle utilisant les données bruitées en vert.

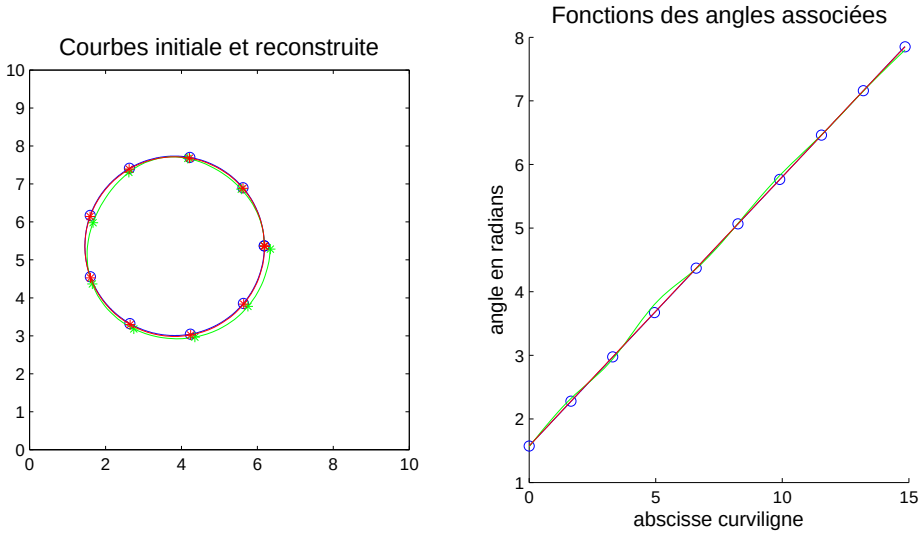


FIG. 1.15 – Reconstruction bruitée d'un cercle, *(bleu) : courbes initiales, (rouge) : courbes reconstruites à partir de données non bruitées, (vert) : courbes reconstruites à partir de données bruitées*

Nous allons appliquer toute une série de tests avec diverses valeurs de bruits, et un nombre divers de capteurs. Nous faisons varier l'amplitude du bruit pour 6, 10 et 15 capteurs pour les bruits suivants : $\{1°, 2°, 4°, 6°, 8°, 10°, 15°\}$. Il y a donc 3 courbes d'erreurs pour des bruits allant de $\pm 1°$ à $\pm 15°$ (voir FIG.1.16).

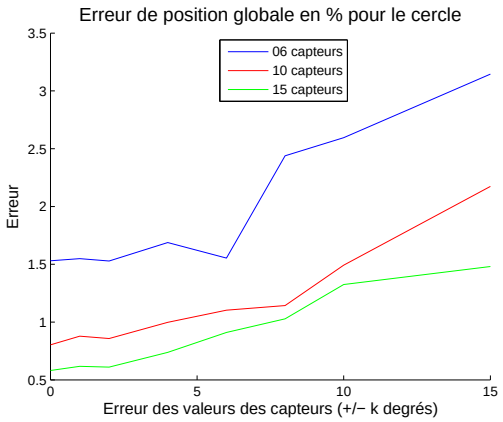


FIG. 1.16 – Erreur en fonction des erreurs des valeurs capteurs pour le cercle

Test 2 : Spline à faible variation Voyons tout d'abord un résultat de courbe obtenu avec 9 capteurs et des valeurs de capteurs connues à $\pm 10°$ près (voir FIG.1.17) : l'erreur de reconstruction obtenue est de 1.82%.

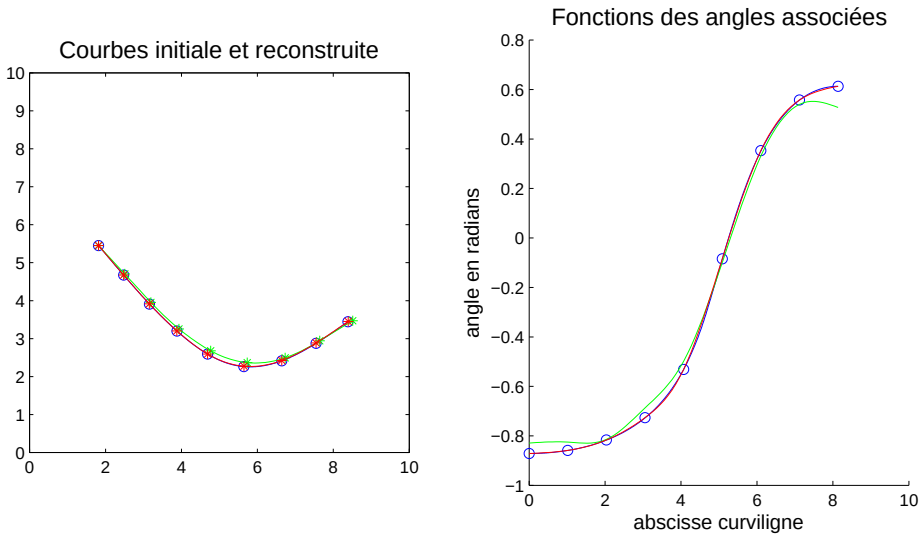


FIG. 1.17 – Reconstruction bruitée d'une spline, *(bleu) : courbes initiales, (rouge) : courbes reconstruites à partir de données non bruitées, (vert) : courbes reconstruites à partir de données bruitées*

Nous faisons les mêmes tests pour cette courbe que pour le cercle avec 5, 9 et 16 capteurs (voir FIG.1.18).

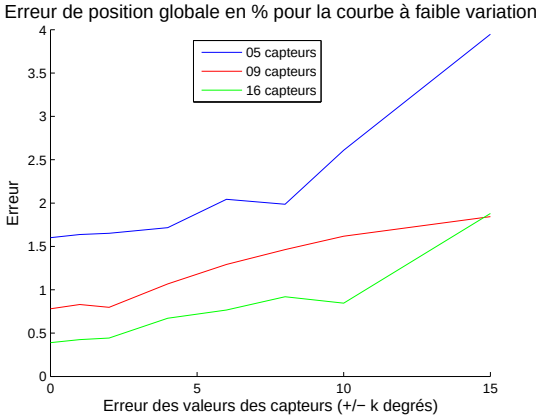


FIG. 1.18 – Erreur en fonction des erreurs des valeurs capteurs pour la spline

Commentaires Ces courbes montrent plusieurs éléments : tout d'abord le fait que les erreurs croissent en fonction des erreurs des valeurs capteurs, mais restent très acceptables. D'autre part, nous voyons que si nous augmentons le nombre de capteurs, les erreurs diminuent. Donc nous pouvons en conclure que nous pouvons compenser l'erreur de précision des capteurs par le nombre de ces capteurs (les erreurs sont compensées par le nombre d'informations). Nous pouvons ainsi considérer que notre méthode de reconstruction est acceptable pour de faibles bruits et un nombre faible de capteurs, mais que si nous dégradons la valeur des capteurs, nous pourrons obtenir des résultats acceptables en augmentant le nombre de capteurs.

b. Bruiter les positions des capteurs Nous allons à présent bruiter la position des capteurs le long de la courbe, ces erreurs pouvant représenter les petites imprécisions de montage lors de l'intégration des capteurs dans le matériau. Nous allons bruiter ces positions avec un bruit blanc gaussien de moyenne nulle et de variance variable. Les données sont précises à $\pm k\%$ près signifient que la variance du bruit utilisé est de $\frac{k}{3L}$, où L est la longueur totale du ruban.

Nous faisons les tests sur les 2 mêmes courbes que précédemment.

Test 1 Voyons tout d'abord un résultat de courbe obtenu avec 10 capteurs et des positions de capteurs connues à $\pm 10\%$ près (voir FIG.1.19) : l'erreur de reconstruction obtenue est de 1.87%.

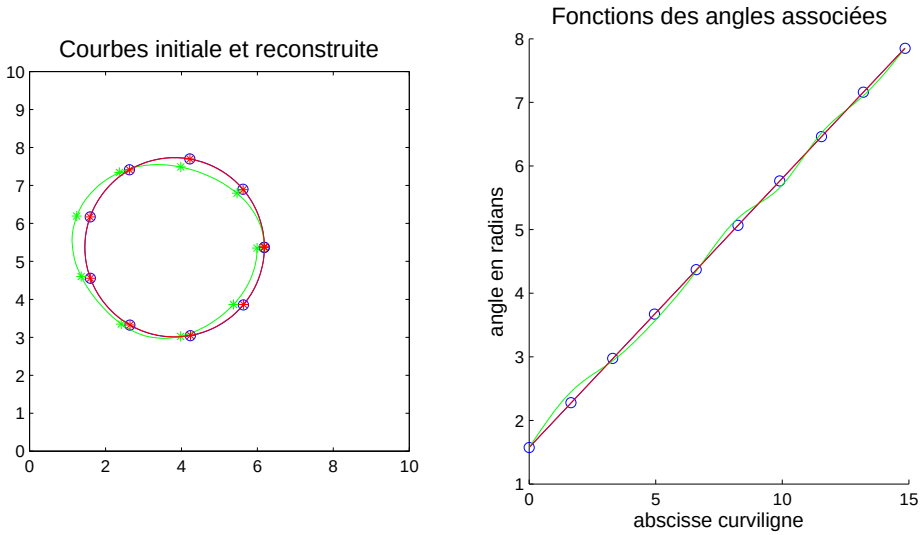


FIG. 1.19 – Reconstruction bruitée d'un cercle, *(bleu) : courbes initiales, (rouge) : courbes reconstruites à partir de données non bruitées, (vert) : courbes reconstruites à partir de données bruitées*

Voici les résultats que nous obtenons pour la batterie de tests suivants : nous faisons varier l'amplitude du bruit des positions pour 6, 10 et 15 capteurs pour les bruits de positions suivants : $\{1\%, 2\%, 4\%, 6\%, 8\%, 10\%, 15\%\}$. Il y a donc 3 courbes d'erreurs pour des bruits allant de $\pm 1\%$ à $\pm 15\%$ (voir FIG.1.20).

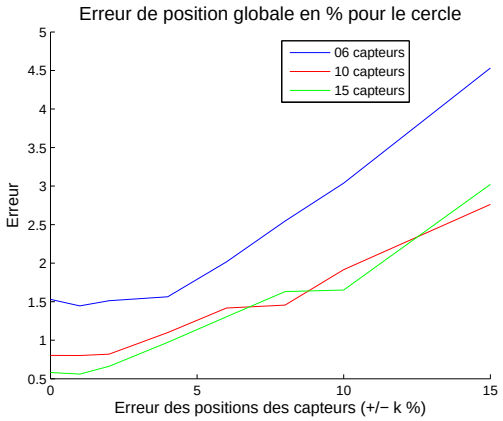


FIG. 1.20 – Erreur en fonction des erreurs des positions des capteurs pour le cercle

Test 2 Voyons tout d'abord un résultat de courbe obtenu avec 9 capteurs et des positions de capteurs connues à $\pm 10\%$ près (voir FIG.1.21) : l'erreur de reconstruction obtenue est de 1.92%.

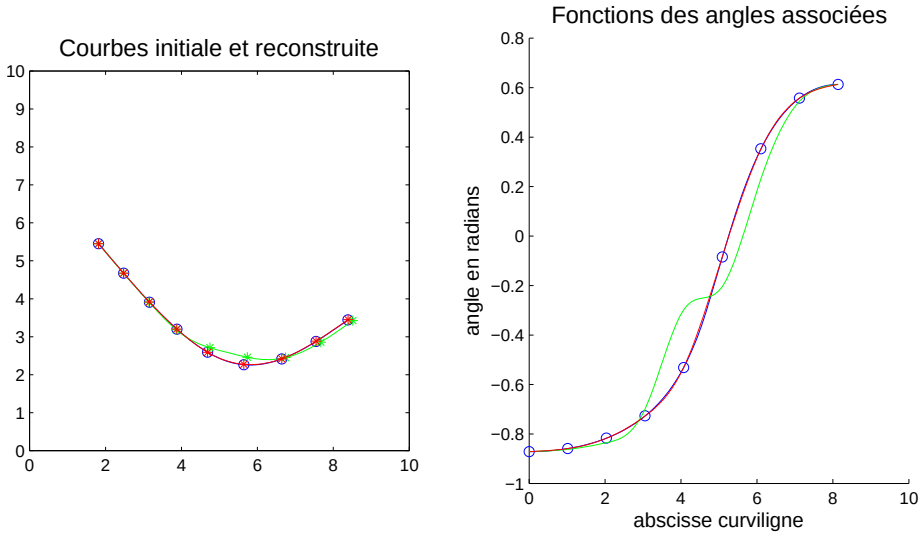


FIG. 1.21 – Reconstruction bruitée d'une spline, *(bleu) : courbes initiales, (rouge) : courbes reconstruites à partir de données non bruitées, (vert) : courbes reconstruites à partir de données bruitées*

Nous faisons les mêmes tests pour cette courbe que pour le cercle avec 5, 9 et 16 capteurs (voir FIG.1.22).

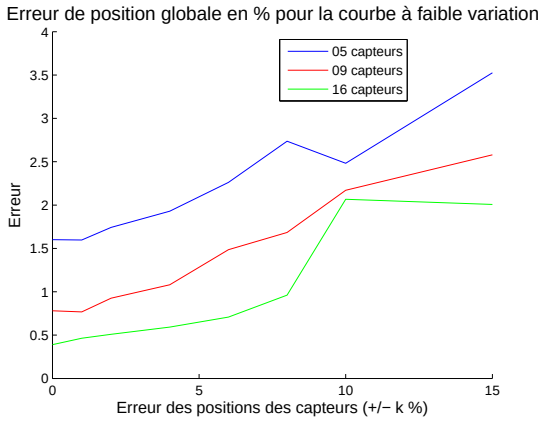


FIG. 1.22 – Erreur en fonction des erreurs des positions des capteurs pour la spline

Commentaires Nous aboutissons au même genre de résultats que précédemment : lorsqu'il y a beaucoup de capteurs, les erreurs sont moindres (la quantité d'informations compense la qualité de ces données), et pour les erreurs effectives que nous aurons avec les vrais systèmes, les erreurs sont acceptables.

1.1.7 CONCLUSION

Dans cette partie, nous avons élaboré une méthode de reconstruction de notre problème, qui consistait à trouver une courbe solution qui avait pour contraintes des directions de tangentes en certains points de la courbe donc nous connaissions la répartition, c'est-à-dire les distances curvilignes entre chaque point de mesure. La méthode élaborée a utilisé la paramétrisation curviligne qui incluait ainsi directement dans le modèle les contraintes de longueur inter-capteurs et ne demandait alors plus qu'à interpoler les angles des tangentes. Nous avons ensuite caractérisé notre méthode : elle apparait avoir une cohérence physique (invariance par rotation, par homothétie, minimisation de la variation de courbure), avoir également des résultats de convergence en fonction du nombre de capteurs, et être robuste au bruit (que ce soit les valeurs des angles des capteurs, ou leur positionnement). Nous allons à présent voir comment nous pouvons nous inspirer de ce travail pour aller plus loin : la reconstruction des courbes gauches.

1.2 COURBES GAUCHES

1.2.1 MODÉLISATION

Nous devons à présent reconstruire des courbes de l'espace avec les mêmes contraintes que précédemment, c'est-à-dire que nous devons reconstruire une courbe dans l'espace en connaissant les orientations des vecteurs tangents en certains points de la courbe. Les données que nous avons sont la répartition des capteurs le long de la courbe, et l'orientation des capteurs à ces endroits-là. Nous allons donc paramétrer la courbe par l'abscisse curviligne ($s \in [0, L]$) tout comme pour les courbes planes :

$$C(s) = \begin{pmatrix} x(s) \\ y(s) \\ z(s) \end{pmatrix}.$$

Sa dérivée est composée de vecteurs unitaires :

$$T(s) = C'(s) = \begin{pmatrix} x'(s) \\ y'(s) \\ z'(s) \end{pmatrix} \text{ tel que } \left\|C'(s)\right\| = 1, \ \forall s.$$

Les données que nous avons sont $(i = 0, ..., n)$:

$$T_i = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} = C'(s_i) \text{ avec } s_i \text{ connus entre 0 et } L \text{ longueur totale de la courbe.}$$

Il est ainsi possible d'étendre la méthode étudiée précédemment sur les courbes planes, qui consiste à reconstruire la fonction dérivée, puis à intégrer pour retrouver la courbe initiale.

L'ensemble des vecteurs dérivés doit rester sur la sphère unité car nous sommes en paramétrisation curviligne. Nous allons nous servir de cette contrainte pour la reconstruction.

1.2.2 PREMIÈRE IDÉE

A. EXPLICATION

Une reconstruction possible du problème en courbes gauches est l'extension directe de la méthode des courbes planes : la paramétrisation en coordonnées sphériques des vecteurs tangents unitaires :

$$T_i = \begin{pmatrix} x_i = \sin(\alpha_i)\cos(\theta_i) \\ y_i = \sin(\alpha_i)\sin(\theta_i) \\ z_i = \cos(\alpha_i) \end{pmatrix}. \tag{1.27}$$

Ainsi, la première idée est d'interpoler les 2 fonctions angles avec des splines cubiques, nous déterminons ainsi la fonction dérivée, puis nous intégrons pour obtenir la courbe finale.

B. DÉFAUTS PRINCIPAUX

Cette méthode peut donner de bons résultats, cependant, 2 problèmes interviennent :

- la reconstruction n'est pas invariante par rotation de la courbe à obtenir (pour 2 courbes identiques obtenues par rotation l'une de l'autre, les reconstructions ne sont pas obtenues par rotation l'une de l'autre...).
- nous n'avons pas la cohérence physique concernant la courbure et la torsion comme pour le cas $2D$.

En effet, les formules de courbure κ et de torsion τ en fonction des 2 angles α et θ sont :

$$\kappa = \sqrt{\alpha'^2 + \theta'^2 \sin^2 \alpha} \tag{1.28a}$$

$$\tau = \frac{2\alpha'^2\theta' \cos\alpha + \alpha'\theta'' \sin\alpha - \theta' \sin\alpha \left(\alpha'' + \theta'^2 \cos\alpha \sin\alpha\right)}{\alpha'^2 + \theta'^2 \sin^2 \alpha} \tag{1.28b}$$

(voir calculs en annexe B). Ces formules ne permettent pas de définir simplement le lien entre les minimisations intervenant dans la méthode (minimisation de $\int \alpha''^2$ et $\int \theta''^2$) et la minimisation de quantités faisant intervenir la courbure et la torsion (κ et τ).

Nous allons donc nous pencher vers une nouvelle méthode, qui reconstruit directement les vecteurs tangents unitaires.

1.2.3 MÉTHODE ÉLABORÉE

Nous devons donc reconstruire la fonction de vecteurs unitaires $T(s)$ pour $s \in [0, L]$ en connaissant les vecteurs unitaires $T(s_i) = T_i = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix}$, avec s_i connus.

Cela revient à chercher une courbe tracée sur la sphère unité, qui interpole un ensemble de points donnés. Nous allons ainsi chercher à interpoler la courbe dérivée par une spline sur la sphère.

La théorie des splines sur la sphère est assez peu étudiée. Il existe d'une part des articles très théoriques, étudiant surtout l'existence de courbes sur des surfaces ayant des propriétés de minimisation d'énergie (cf. [HP04a], [HP04b], [PH05] et [PH03]), mais qui n'apportent pas de méthode de construction. D'autre part, il existe une autre catégorie d'articles qui sont au contraire plus applicatifs, dont le but est d'interpoler dans l'espace des quaternions (donc sur la sphère de $\mathbb{R}^4$) afin de générer des mouvements lisses d'objets (cf. [SHO85]et [NIE04]). Ces papiers sont alors des extensions des B-splines du plan, qui sont appelées splines sur la sphère par analogie.

Nous utilisons ici les splines sur la sphère telles qu'elles ont été développées dans [NIE04]. Voyons dans un premier temps un rappel sur les splines cubiques en espace plan, puis nous verrons quelles modifications nous devons appliquer pour créer les splines sur la sphère.

A. RAPPEL SPLINES SUR LE PLAN

A.a. Cubiques de Bézier et interprétation géométrique Nous savons que les courbes cubiques peuvent s'écrire en fonction d'un polygone de contrôle $P_i, i = 0, ..., 3$ et de la base de fonctions polynomiales de Bernstein :

$$B(t) = \sum_{i=0}^{3} P_i C_i^3 (1-t)^{3-i} t^i.$$

Nous pouvons aussi évaluer la valeur de cette courbe en un point en fonction du polygone de contrôle et d'interpolations linéaires successives (algorithme de DeCasteljau) (voir FIG.1.23).

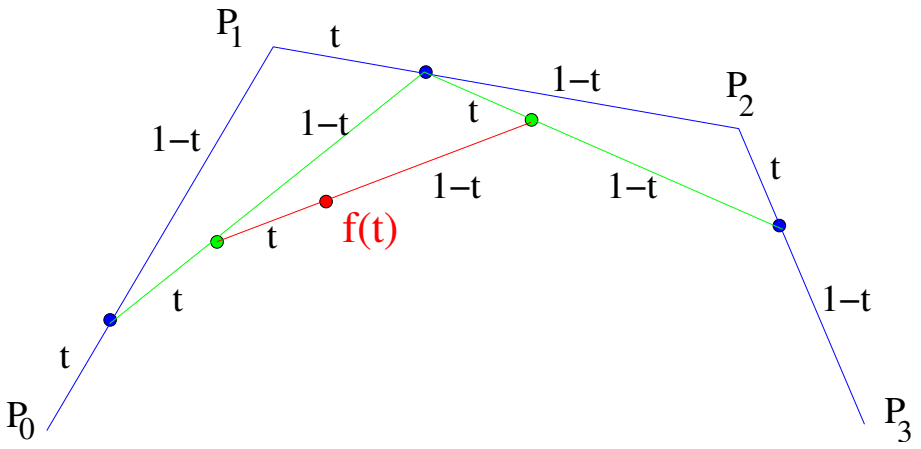


FIG. 1.23 – Algorithme de DeCasteljau

Nous notons cette interpolation $BC[P_0, P_1, P_2, P_3](t)$.

Il est maintenant possible de créer des splines cubiques en raccordant plusieurs cubiques, c'est ce que nous appelons les B-splines.

A.b. Les B-splines Etant donnés les points de contrôle d_i et les noeuds s_i, nous définissons la spline cubique suivante (voir FIG.1.24) :

$$C(s) = BC[T_i, R_i, L_{i+1}, T_{i+1}]\left(\frac{s - s_i}{h_{i+1}}\right) \tag{1.29}$$

pour $s_i \leq s \leq s_{i+1}, i = 0, ..., n-1$, $h_i = s_i - s_{i-1}$, avec :

$$R_i = (1 - \mu_i)d_i + \mu_i d_{i+1} \tag{1.30a}$$
$$L_{i+1} = (1 - \lambda_{i+1})d_i + \lambda_{i+1} d_{i+1} \tag{1.30b}$$
$$T_i = (1 - \delta_i)L_i + \delta_i R_i \tag{1.30c}$$

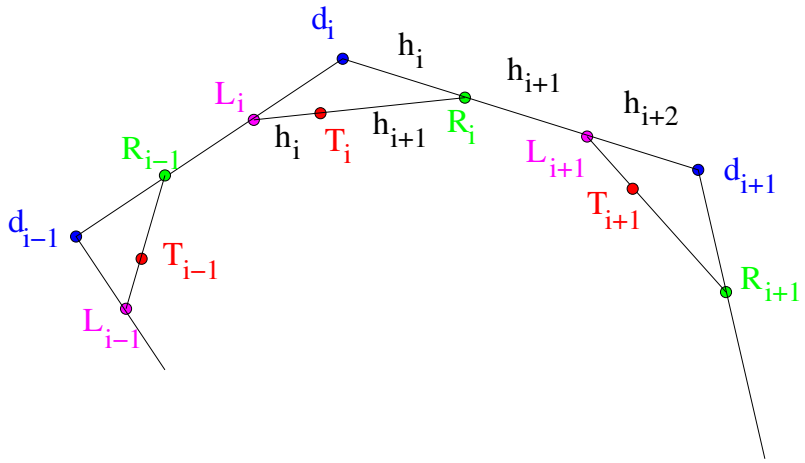


FIG. 1.24 – Construction du polygone de contrôle de la B-spline

$$\lambda_i = \frac{h_{i-1}+h_i}{h_{i-1}+h_i+h_{i+1}} \tag{1.31a}$$

$$\mu_i = \frac{h_i}{h_i+h_{i+1}+h_{i+2}} \tag{1.31b}$$

$$\delta_i = \frac{h_i}{h_i+h_{i+1}} \tag{1.31c}$$

La spline interpole les points T_i. Pour la déterminer, nous devons donc déterminer les points de contrôle d_i. Avec les formules précédentes, nous aboutissons à un système linéaire :

$$\begin{pmatrix} 1 & 0 & \cdots & \cdots & \cdots & 0 \\ a_1 & b_1 & c_1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & \cdots & \cdots & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} {}^t d_0 \\ {}^t d_1 \\ \vdots \\ \vdots \\ \vdots \\ {}^t d_n \end{pmatrix} = \begin{pmatrix} {}^t T_0 \\ {}^t T_1 \\ \vdots \\ \vdots \\ \vdots \\ {}^t T_n \end{pmatrix} \tag{1.32}$$

$$a_i = (1-\delta_i)(1-\lambda_i) \tag{1.33a}$$

$$b_i = (1-\delta_i)\lambda_i + \delta_i(1-\mu_i) \tag{1.33b}$$

$$c_i = \delta_i \mu_i \tag{1.33c}$$

La courbe est G^2 par définition.

Voyons à présent comment nous pouvons appliquer ceci pour obtenir des courbes sur la sphère.

B. LES B-SPLINES SUR LA SPHÈRE

Cette section détaille la démarche et l'algorithme de [NIE04].

B.a. Comment passer sur la sphère Il nous suffit de remplacer les interpolations linéaires par des interpolations jouant le même rôle sur la sphère unité : les interpolations sphériques linéaires : slerp (spherical linear interpolation).

En espace plan, l'interpolation linéaire est :

$$P = (1-t)A + tB.$$

Sur la sphère, cela est remplacé par :

$$Slerp(A,B,t) = \frac{\sin((1-t)\theta)A + \sin(t\theta)B}{\sin\theta}, \tag{1.34}$$

avec θ l'angle entre le vecteur A et le vecteur B : $\theta = \cos^{-1}(A,B)$ (voir FIG.1.25).

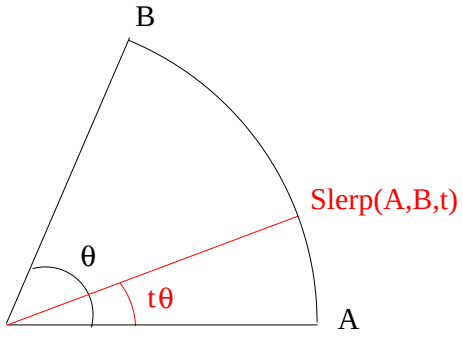


FIG. 1.25 – Interpolation sphérique

Notons que $\forall t$, nous avons $\|Slerp(A,B,t)\| = 1$. En effet :

$$\begin{aligned}
\|Slerp(A,B,t)\|^2 &= \left\langle \frac{\sin((1-t)\theta)A + \sin(t\theta)B}{\sin\theta}, \frac{\sin((1-t)\theta)A + \sin(t\theta)B}{\sin\theta} \right\rangle \\
&= \frac{1}{\sin^2\theta}\left(\sin^2((1-t)\theta)\langle A,A\rangle + \sin^2 t\theta\langle B,B\rangle + 2\sin((1-t)\theta)\sin t\theta\langle A,B\rangle\right) \\
&= \frac{\sin^2((1-t)\theta) + \sin^2 t\theta + 2\sin((1-t)\theta)\sin t\theta}{\sin^2\theta}
\end{aligned}$$

En développant $\sin((1-t)\theta)$, nous obtenons les simplifications voulues.

Ainsi, nous définissons les fonctions cubiques sur la sphère par 4 points de contrôle sur la sphère et le modèle étendu de DeCasteljau (en remplaçant les interpolations linéaires par les interpolations sphériques linéaires). La cubique sphérique sera notée $SC[P_0,P_1,P_2,P_3](s)$.

B.b. Algorithme Nous allons maintenant définir les points de contrôle de la spline en utilisant la même méthode que pour les courbes dans le plan, mais nous utilisons ici les interpolations sphériques.

Etant donnés les points de contrôle d_i (qui sont sur la sphère unité) et les noeuds s_i, nous définissons la spline cubique sphérique suivante :

$$T(s) = SC[T_i, R_i, L_{i+1}, T_{i+1}]\left(\frac{s-s_i}{h_{i+1}}\right) \tag{1.35}$$

pour $s_i \leq s \leq s_{i+1}, i = 0, ..., n-1$, $h_i = s_i - s_{i-1}$, avec :

$$R_i = Slerp(d_i, d_{i+1}, \mu_i) \tag{1.36a}$$
$$L_{i+1} = Slerp(d_i, d_{i+1}, \lambda_{i+1}) \tag{1.36b}$$
$$T_i = Slerp(L_i, R_i, \delta_i) \tag{1.36c}$$

$$\lambda_i = \frac{h_{i-1} + h_i}{h_{i-1} + h_i + h_{i+1}}$$
$$\mu_i = \frac{h_i}{h_i + h_{i+1} + h_{i+2}}$$
$$\delta_i = \frac{h_i}{h_i + h_{i+1}}$$

La spline interpole des points T_i. Nous devons déterminer les points de contrôle d_i. Contrairement à ce que nous obtenons en espace plan, le système à résoudre n'est pas linéaire :

$$\begin{cases} T_0 = d_0 \\ T_i = \dfrac{1}{\sin\beta_i} \left(\begin{array}{l} \left(\dfrac{\sin((1-\delta_i)\beta_i)\sin((1-\lambda_i)\alpha_i)}{\sin\alpha_i} \right) d_{i-1} \\ + \left(\dfrac{\sin((1-\delta_i)\beta_i)\sin(\lambda_i\alpha_i)}{\sin\alpha_i} + \dfrac{\sin(\delta_i\beta_i)\sin((1-\mu_i)\alpha_{i+1})}{\sin(\alpha_{i+1})} \right) d_i \\ + \left(\dfrac{\sin(\delta_i\beta_i)\sin(\mu_i\alpha_{i+1})}{\sin(\alpha_{i+1})} \right) d_{i+1} \end{array} \right) \\ T_n = d_n \end{cases} \tag{1.37}$$

où :

$$\alpha_i = \cos^{-1}(d_{i-1}, d_i)$$
$$\beta_i = \cos^{-1}(L_i, R_i)$$

Nous allons donc résoudre ce système avec une méthode itérative.

En voici l'algorithme :

1. déterminer les points de contrôle initiaux $(d_i^{(0)}, i = 0, ..., n)$ (par exemple les valeurs normalisées de la spline 3D),
2. déterminer les valeurs des angles (α_i, β_i) en fonction de ces points de contrôle,
3. calculer les nouveaux points de contrôle avec :

$$d_0^{(k)} = T_0$$
$$d_i^{(k)} = \frac{T_i \sin\beta_i - \left(\dfrac{\sin((1-\delta_i)\beta_i)\sin((1-\lambda_i)\alpha_i)}{\sin\alpha_i} \right) d_{i-1}^{(k-1)} - \left(\dfrac{\sin(\delta_i\beta_i)\sin(\mu_i\alpha_{i+1})}{\sin(\alpha_{i+1})} \right) d_{i+1}^{(k-1)}}{\dfrac{\sin((1-\delta_i)\beta_i)\sin(\lambda_i\alpha_i)}{\sin\alpha_i} + \dfrac{\sin(\delta_i\beta_i)\sin((1-\mu_i)\alpha_{i+1})}{\sin(\alpha_{i+1})}}$$
$$d_n^{(k)} = T_n$$

et :

$$\begin{aligned} R_i &= (1-\mu_i)d_i^{(k-1)} + \mu_i d_{i+1}^{(k-1)} \\ L_{i+1} &= (1-\lambda_{i+1})d_i^{(k-1)} + \lambda_{i+1} d_{i+1}^{(k-1)} \\ \alpha_i &= \cos^{-1}(d_{i-1}^{(k-1)}, d_i^{(k-1)}) \\ \beta_i &= \cos^{-1}(L_i, R_i) \end{aligned}$$

4. normaliser les d_i pour qu'ils soient vraiment sur la sphère (en effet, comme les interpolations sphériques utilisent les angles α_i et β_i qui sont les valeurs des angles à l'étape précédente, nous obtenons des d_i qui ne sont pas exactement sur la sphère unité),
5. itérer jusqu'à convergence avec le critère suivant : que les $T_i^{(k)}$ calculés avec le polygone de contrôle des $(d_i^{(0)}, i = 0, ..., n)$ soient proches des T_i à obtenir :

$$\sqrt{\frac{\sum_{i=0}^{n} \left\| T_i^{(k)} - T_i \right\|^2}{\sum_{i=0}^{n} \|T_i\|^2}} \leq Tol$$

L'algorithme de [NIE04] est modifié par l'ajout de l'étape 4 dans laquelle nous normalisons les points de contrôle. L'oubli de cette étape rend alors tous les calculs erronés (car les calculs prennent pour hyptohèse que nous travaillons avec des vecteurs unitaires) et amène alors notamment des problèmes de convergence de la méthode. Nous avons aussi modifié le critère d'arrêt de l'algorithme. Alors que [NIE04] propose un arrêt lorsque les points de contrôle convergent vers un vecteur solution, nous proposons un arrêt lorsque les points que nous interpolons sont proches de ceux que nous cherchons à interpoler (qui est le but de notre algorithme).

1.2.4 QUELQUES COURBES

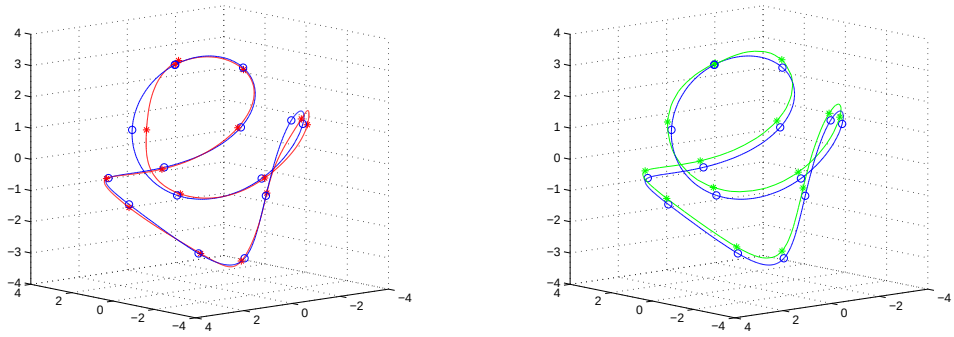
Nous utilisons les mêmes erreurs que pour les courbes planes pour caractériser les résultats :

- erreur sur la longueur de la courbe reconstruite,
- erreur sur la position des capteurs,
- erreur sur la position du point d'arrivée,
- erreur de positionnement global de la courbe,
- erreur de Hausdorff modifiée.

Nous allons voir ici quelques résultats obtenus avec les deux méthodes. Dans les figures suivantes, les courbes bleues représentent les courbes initiales à reconstruire, et les courbes rouges et vertes sont les courbes obtenues par les méthodes de reconstruction, les courbes rouges pour les reconstructions des deux angles indépendamment, et les courbes vertes par reconstruction directe des vecteurs tangents sur la sphère.

TEST 1 : COURBE FERMÉE

Nous testons les deux méthodes sur une courbe fermée sur laquelle nous avons placé 15 points de mesure. Les deux courbes résultats sont sur la figure FIG.1.26. Le graphe de gauche représente en bleu la courbe initiale et en rouge sa reconstruction avec la méthode qui interpole les angles de la paramétrisation sphèrique indépendamment ; le graphe de droite superpose à la courbe initiale bleue la courbe verte qui représente la reconstruction faisant intervenir l'interpolation sur la sphère de la dérivée.



(a) Reconstruction par les angles

(b) Reconstruction sur la sphère

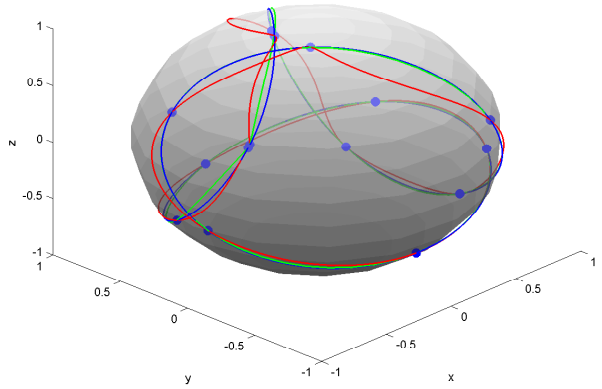
FIG. 1.26 – Courbe fermée : Courbes reconstruites, *(bleu) : courbes initiales, (rouge) et (vert) : courbes reconstruites selon les deux méthodes*

Ces courbes sont le résultat de l'intégration de leur courbe dérivée associée. Ces courbes dérivées qui sont des courbes sur la sphère sont visualisées dans la figure FIG.1.27, où nous voyons en (a) la superposition de la dérivée initiale (bleu), la dérivée par la méthode 1 (rouge) et la dérivée par la méthode 2 (vert). Sur le graphe (b), nous avons seulement fait apparaître la dérivée initiale et celle de la méthode 1, et sur le graphe (c) la dérivée initiale et la dérivée de la méthode 2.

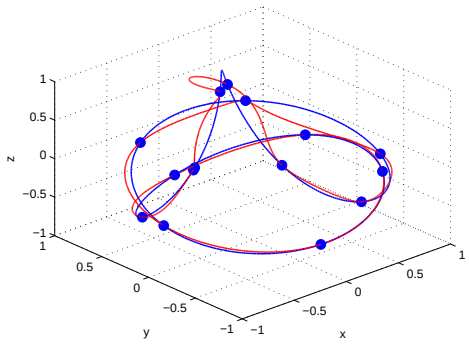
Les erreurs obtenues sont dans le tableau TAB.1.5.

Erreurs	Reconstruction des angles indépendants	Reconstruction sur la sphère
e_L	$5.15.10^{-4}\%$	0.15%
e_{capt}	0.55%	0.56%
e_{fin}	0.96%	0.18%
e_{pos}	0.47%	0.87%
e_{Haus}	1.27%	0.83%

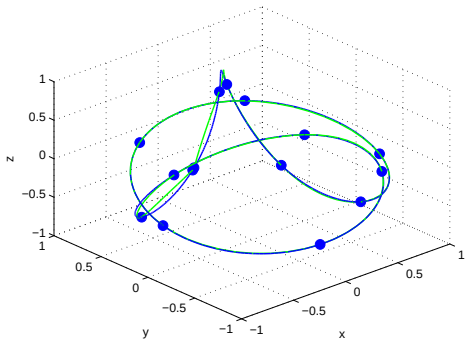
TAB. 1.5 – Courbe fermée : Erreurs de reconstruction



(a) Reconstructions



(b) Reconstruction par les angles



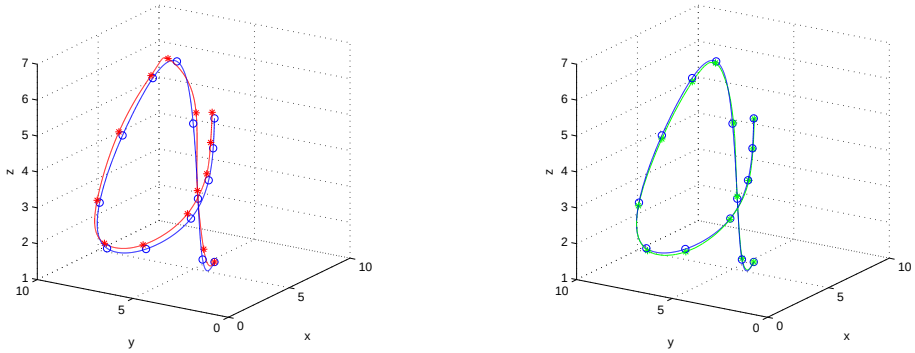
(c) Reconstruction sur la sphère

FIG. 1.27 – Courbe fermée : Courbes des vecteurs tangents sur la sphère, *(bleu) : courbes initiales, (rouge) et (vert) : courbes reconstruites selon les deux méthodes*

Les résultats visuels montrent clairement un meilleur résultat en ce qui concerne la reconstruction de la courbe dérivée avec la méthode d'interpolation directe sur la sphère, en effet, reconstruire les angles de façon indépendante donne des courbes sur la sphère vraiment irrégulières. De plus, les erreurs sont similaires en ce qui concerne la position des capteurs, et meilleurs pour la position du point d'arrivée et l'erreur de Hausdorff, mais une erreur plus élevée de position globale. Donc nous pouvons dire que la méthode de reconstruction directement sur la sphère donne une courbe globalement plus régulière et plus proche que l'autre méthode (erreur de Hausdorff).

TEST 2 : SPLINE 3D

La deuxième courbe est reconstruite à partir de 14 points de mesure répartis équitablement. Les deux courbes résultats sont sur la figure FIG.1.28. Les reconstructions des



(a) Reconstruction par les angles

(b) Reconstruction sur la sphère

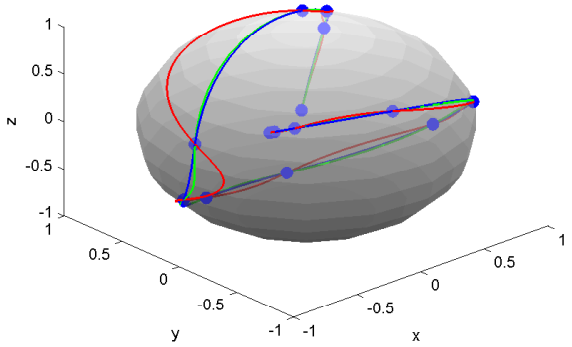
FIG. 1.28 – Spline 3D : Courbes reconstruites, *(bleu) : courbes initiales, (rouge) et (vert) : courbes reconstruites selon les deux méthodes*

courbes de vecteurs tangents sont visualisées dans la figure FIG.1.29.

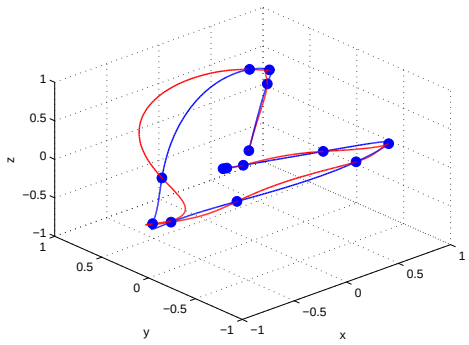
Les erreurs obtenues sont dans le tableau TAB.1.6.

Erreurs	Reconstruction des angles indépendants	Reconstruction sur la sphère
e_L	$3.23.10^{-4}\%$	0.08%
e_{capt}	0.84%	0.50%
e_{fin}	0.66%	0.74%
e_{pos}	0.83%	0.80%
e_{Haus}	1.52%	0.93%

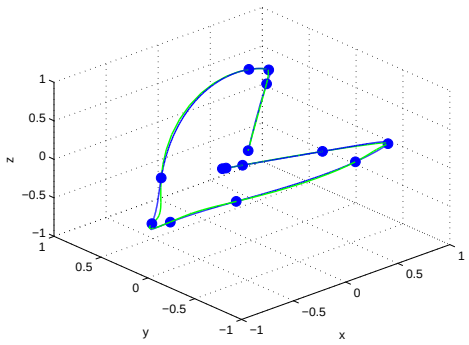
TAB. 1.6 – Spline 3D : Erreurs de reconstruction



(a) Reconstructions



(b) Reconstruction par les angles



(c) Reconstruction sur la sphère

FIG. 1.29 – Spline 3D : Courbes des vecteurs tangents sur la sphère, *(bleu) : courbes initiales, (rouge) et (vert) : courbes reconstruites selon les deux méthodes*

Sur cet exemple, nous observons visuellement et avec les erreurs calculées que la méthode de reconstruction de la dérivée directement sur la sphère est meilleure que la premiere méthode explorée.

Nous allons à présent caractériser cette méthode en étudiant ses propriétés de consistance et de convergence.

1.2.5 CONSISTANCE

A. SENS PHYSIQUE

Voyons la conséquence sur la courbe des contraintes de reconstruction de la courbe dérivée. Nous reconstruisons la fonction dérivée en utilisant une spline cubique sur la sphère. Nous savons que dans les espaces plans, les splines cubiques minimisent leur énergie. Les splines cubiques sur la sphère ayant été obtenues par analogie, nous pouvons supposer qu'elles minimisent également leur énergie : voyons alors les conséquences d'une telle hypothèse.

Sous cette hypothèse, notre méthode minimise :

$$\int \left\|T''(s)\right\|^2 ds \tag{1.38}$$

avec T vecteurs tangents à la courbe. Nous cherchons l'influence que cela a sur des données intrinsèques à la courbe (comme par exemple la courbure et la torsion), nous utilisons alors les formules de Serret-Frenet :

$$\begin{aligned} T' &= \kappa N \\ N' &= -\kappa T + \tau B \end{aligned}$$

avec (T, N, B) repère local de Frenet, κ la courbure et τ la torsion. Donc :

$$\begin{aligned} T'' &= \kappa' N + \kappa N' \\ &= \kappa' N + \kappa(-\kappa T + \tau B) \\ &= \kappa' N - \kappa^2 T + \kappa\tau B \\ \left\|T''\right\|^2 &= \left\langle T'', T''\right\rangle \\ &= \kappa'^2 + \kappa^2(\kappa^2 + \tau^2) \end{aligned}$$

Ainsi, en procédant par analogie par rapport aux splines dans des espaces plans, la méthode d'interpolation par splines sur la sphère minimise :

$$\int \kappa'^2 + \kappa^2(\kappa^2 + \tau^2) \tag{1.39}$$

c'est-à-dire que nous minimisons une quantité englobant la torsion, la courbure et la variation de la courbure .

Cette méthode fait donc intervenir la minimisation de valeurs intrinsèques à la courbe contrairement à la première méthode dont nous avions parlé, qui minimisait des quantités dépendant des angles de paramétrisation, donc dépendant de l'orientation de la courbe.

B. INVARIANCE PAR ROTATION

Nous allons ici montrer l'invariance par rotation de la solution obtenue, c'est-à-dire que pour deux ensembles de données obtenus par rotation l'un de l'autre, les courbes reconstruites par cette méthode seront deux courbes identiques à cette rotation près.

Soient deux ensembles de vecteurs unitaires $\{T_i^{(1)}, i = 0, ..., n\}$ et $\{T_i^{(2)}, i = 0, ..., n\}$ obtenus par rotation l'un de l'autre, c'est-à-dire qu'il existe une matrice de rotation $\mathscr{M}_R$ telle que :

$$T_i^{(2)} = \mathscr{M}_R T_i^{(1)}, i = 0, ..., n. \tag{1.40}$$

Nous devons alors montrer que les courbes $C^{(1)}$ (courbe reconstruite à partir des $T_i^{(1)}$) et $C^{(2)}$ (courbe reconstruite à partir des $T_i^{(2)}$) sont identiques à la rotation $\mathscr{M}_R$ près, c'est-à-dire que :

$$C^{(2)}(s) = \mathscr{M}_R C^{(1)}(s), \forall s.$$

Deux courbes sont identiques à une rotation près signifie que leur dérivée sont également obtenues par rotation l'une de l'autre :

$$C'^{(2)}(s) = \mathscr{M}_R C'^{(1)}(s), \forall s.$$

Etant donné que les courbes dérivées sont définies de façon unique grâce à leur polygone de contrôle, prouver l'égalité précédente revient exactement à prouver que les polygones de contrôle sont obtenus par rotation l'un de l'autre : (pour $\{d_i^{(1)}, i = 0, ..., n\}$ polygone associé à la courbe $C'^{(1)}$ et $\{d_i^{(2)}, i = 0, ..., n\}$ polygone associé à la courbe $C'^{(2)}$), nous devons prouver :

$$d_i^{(2)} = \mathscr{M}_R d_i^{(1)}, i = 0, ..., n. \tag{1.41}$$

Les polygones de contrôle sont déterminés par une initialisation et des itérations. Nous allons donc prouver l'égalité (1.41) par récurrence.

B.a. Initialisation L'étape d'initialisation de l'algorithme consiste à calculer la spline 3D passant par les points. Ce calcul consiste à la résolution d'un systéme linéaire, dont la matrice est :

$$M = \begin{pmatrix} 1 & 0 & \cdots & \cdots & \cdots & 0 \\ a_1 & b_1 & c_1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & \cdots & \cdots & \cdots & 0 & 1 \end{pmatrix} \text{ avec } \begin{array}{l} a_i = (1-\delta_i)(1-\lambda_i) \\ b_i = (1-\delta_i)\lambda_i + \delta_i(1-\mu_i) \\ c_i = \delta_i \mu_i \end{array} \tag{1.42}$$

avec

$$\begin{aligned} \lambda_i &= \frac{h_{i-1} + h_i}{h_{i-1} + h_i + h_{i+1}} \\ \mu_i &= \frac{h_i}{h_i + h_{i+1} + h_{i+2}} \\ \delta_i &= \frac{h_i}{h_i + h_{i+1}} \end{aligned}$$

Donc la matrice est la même pour la résolution des 2 systèmes car ne dépend que des distances inter-capteurs. Nous avons ainsi :

$$M.\begin{pmatrix} {}^t d_0^{(1),0} \\ \vdots \\ {}^t d_n^{(1),0} \end{pmatrix} = \begin{pmatrix} {}^t T_0^{(1)} \\ \vdots \\ {}^t T_n^{(1)} \end{pmatrix}$$

et d'après (1.40), nous pouvons relier les deux polygones de contrôle :

$$\begin{aligned} \begin{pmatrix} {}^t d_0^{(2),0} \\ \vdots \\ {}^t d_n^{(2),0} \end{pmatrix} &= M^{-1}\begin{pmatrix} {}^t T_0^{(2)} \\ \vdots \\ {}^t T_n^{(2)} \end{pmatrix} = M^{-1}\begin{pmatrix} {}^t\left(\mathscr{M}_R T_0^{(1)}\right) \\ \vdots \\ {}^t\left(\mathscr{M}_R T_n^{(1)}\right) \end{pmatrix} \\ &= M^{-1}\begin{pmatrix} {}^t T_0^{(1)} \\ \vdots \\ {}^t T_n^{(1)} \end{pmatrix} {}^t\mathscr{M}_R = \begin{pmatrix} {}^t d_0^{(1),0} \\ \vdots \\ {}^t d_n^{(1),0} \end{pmatrix} {}^t\mathscr{M}_R \\ &= \begin{pmatrix} {}^t\left(\mathscr{M}_R d_0^{(1),0}\right) \\ \vdots \\ {}^t\left(\mathscr{M}_R d_n^{(1),0}\right) \end{pmatrix} \end{aligned}$$

Nous avons alors le résultat souhaité, c'est-a-dire :

$$d_i^{(2),0} = \mathscr{M}_R d_i^{(1),0}, i = 0, ..., n. \tag{1.43}$$

B.b. Récurrence Supposons que l'égalité des polygones soit vraie au rang $k-1$. Prouvons le au rang k. Nous avons :

$$d_i^{(2),k-1} = \mathscr{M}_R d_i^{(1),k-1}, i = 0, ..., n. \tag{1.44}$$

La relation de récurrence est la suivante :

$$\begin{aligned} d_0^{(1),k} &= T_0^{(1)} \\ d_i^{(1),k} &= \frac{T_i^{(1)}\sin\beta_i^{(1)} - \left(\frac{\sin((1-\delta_i)\beta_i^{(1)})\sin((1-\lambda_i)\alpha_i^{(1)})}{\sin\alpha_i^{(1)}}\right) d_{i-1}^{(1),k-1} - \left(\frac{\sin(\delta_i\beta_i^{(1)})\sin(\mu_i\alpha_{i+1}^{(1)})}{\sin(\alpha_{i+1}^{(1)})}\right) d_{i+1}^{(1),k-1}}{\frac{\sin((1-\delta_i)\beta_i^{(1)})\sin(\lambda_i\alpha_1^{(1)})}{\sin\alpha_i^{(1)}} + \frac{\sin(\delta_i\beta_i^{(1)})\sin((1-\mu_i)\alpha_{i+1}^{(1)})}{\sin(\alpha_{i+1}^{(1)})}}, \\ & i = 1, ..., n-1 \\ d_n^{(1),k} &= T_n^{(1)} \end{aligned}$$

avec

$$\begin{aligned} \alpha_i^{(1)} &= \cos^{-1}(d_{i-1}^{(1),k-1}, d_i^{(1),k-1}) \\ \beta_i^{(1)} &= \cos^{-1}(L_i^{(1),k-1}, R_i^{(1),k-1}) \\ R_i^{(1),k-1} &= Slerp(d_i^{(1),k-1}, d_{i+1}^{(1),k-1}, \mu_i) \\ L_i^{(1),k-1} &= Slerp(d_{i-1}^{(1),k-1}, d_i^{(1),k-1}, \lambda_i) \end{aligned}$$

et de même pour le polygone des $d_i^{(2),k}$.

Pour obtenir le résultat que nous cherchons, il nous suffit de montrer que :

$$\alpha_i^{(2)} = \alpha_i^{(1)} \quad (1.45a)$$

$$\beta_i^{(2)} = \beta_i^{(1)} \quad (1.45b)$$

Ceci est vrai pour les angles α : en effet, nous avons (1.44), et les angles sont conservés par les rotations. Donc l'angle entre $d_{i-1}^{(1),k-1}$ et $d_i^{(1),k-1}$ $(= \alpha_i^{(1)})$ est le même que l'angle entre $d_{i-1}^{(2),k-1}$ et $d_i^{(2),k-1}$ $(= \alpha_i^{(2)})$.

Pour prouver l'égalité des β_i, cela revient à prouver que des $R_i^{(2),k-1}$ sont obtenus par rotation des $R_i^{(1),k-1}$ et de même, les $L_i^{(2),k-1}$ sont obtenus par rotation des $L_i^{(1),k-1}$ (puis nous utiliserons le fait que les angles ne sont pas modifiés par isométrie). Or :

$$\begin{aligned} L_i^{(2),k-1} &= \frac{\sin((1-\lambda_i)\alpha_i)d_{i-1}^{(2),k-1} + \sin(\lambda_i\alpha_i)d_i^{(2),k-1}}{\sin\alpha_i} \\ &= \frac{\sin((1-\lambda_i)\alpha_i)\mathscr{M}_R d_{i-1}^{(1),k-1} + \sin(\lambda_i\alpha_i)\mathscr{M}_R d_i^{(1),k-1}}{\sin\alpha_i} \\ &= \mathscr{M}_R \frac{\sin((1-\lambda_i)\alpha_i)d_{i-1}^{(1),k-1} + \sin(\lambda_i\alpha_i)d_i^{(1),k-1}}{\sin\alpha_i} \\ &= \mathscr{M}_R . L_i^{(1),k-1} \end{aligned}$$

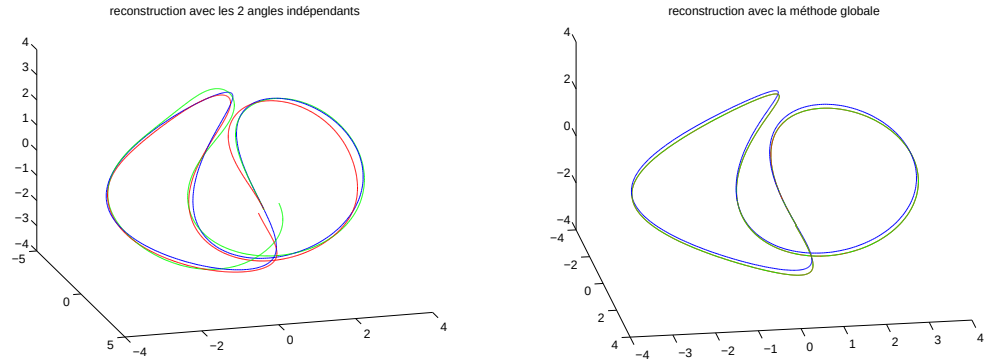
et idem pour $R_i^{(2),k-1}$.

Nous avons ainsi :

$$d_i^{(2),k} = \mathscr{M}_R d_i^{(1),k}, i = 0, ..., n, \forall k \geq 0 \quad (1.46)$$

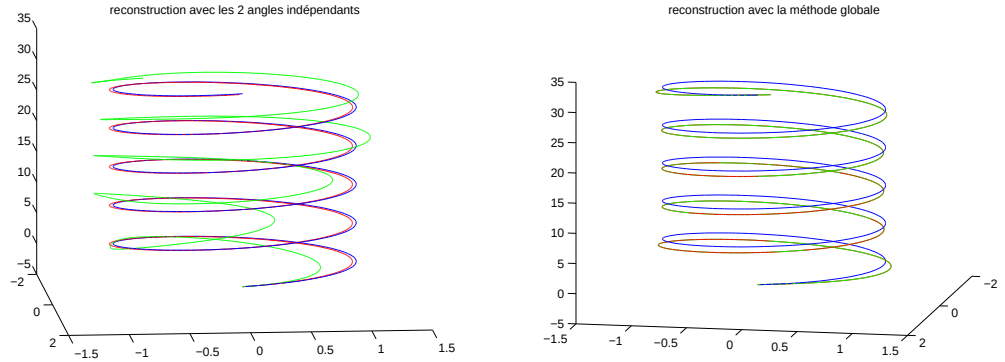
Les polygones de contrôle sont obtenus par rotation l'un de l'autre, donc les courbes des dérivées obtenues sont identiques à une rotation près, donc les courbes finales reconstruites sont identiques à une rotation près. La propriété est démontrée : la méthode donne des courbes invariantes par rotation.

Voyons à présent une illustration de l'invariance par rotation de la méthode sur quelques exemples. Pour les différents exemples suivants (voir les figures FIG.1.30, FIG.1.31 et FIG.1.32), nous allons appliquer les algorithmes de reconstruction à une courbe et son identique par rotation, puis nous appliquerons la rotation inverse à la reconstruction de la courbe ayant subi la rotation afin de comparer les résultats. Nous verrons ainsi le fait que la méthode de reconstruction par paramétrisation sphérique n'est pas invariante par rotation, contrairement à la méthode d'interpolation globale. Les courbes bleues représentent les courbes initiales, les courbes rouges les courbes reconstruites, et les courbes vertes les reconstructions de courbes ayant subi une rotation (et à laquelle nous avons appliqué la rotation inverse afin de les comparer).



(a) Reconstruction par les 2 angles (b) Reconstruction sur la sphère

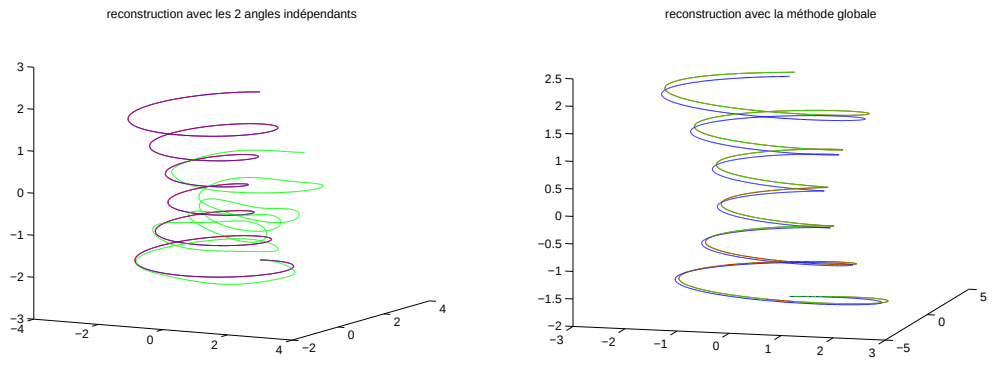
FIG. 1.30 – Courbe fermée : invariance par rotation, *(bleu) : courbes initiales, (rouge) : courbes reconstruites dans cette orientation, (vert) : courbes reconstruites dans une orientation différente et réorientées pour la comparaison*



(a) Reconstruction par les 2 angles (b) Reconstruction sur la sphère

FIG. 1.31 – Courbe hélicoïdale : invariance par rotation, *(bleu) : courbes initiales, (rouge) : courbes reconstruites dans cette orientation, (vert) : courbes reconstruites dans une orientation différente et réorientées pour la comparaison*

Nous voyons clairement les courbes rouges et vertes très distintes dans la colonne de gauche, où nous voyons même de mauvaises reconstructions, alors que sur la droite, les deux courbes sont confondues. Ces 3 exemples nous montrent le fait que la paramétrisation des vecteurs tangents par les coordonnées sphériques rend la reconstruction complètement dépendante de l'orientation de cette courbe par rapport au repère, ce qui n'est pas le cas avec la méthode d'interpolation directe sur la sphère.



(a) Reconstruction par les 2 angles

(b) Reconstruction sur la sphère

FIG. 1.32 – Courbe hélicoïdale à rayon variable : invariance par rotation, *(bleu) : courbes initiales, (rouge) : courbes reconstruites dans cette orientation, (vert) : courbes reconstruites dans une orientation différente et réorientées pour la comparaison*

C. INVARIANCE PAR HOMOTHÉTIE

Nous allons ici nous intéresser à l'invariance par homothétie que fournit cette méthode de reconstruction. Soient deux ensembles de données $\{T_i^{(1)}, i = 0, ..., n\}$ connus aux points $\{s_i, i = 0, ..., n\}$ et $\{T_i^{(2)}, i = 0, ..., n\}$ connus aux points $\{\sigma_i = K.s_i, i = 0, ..., n\}$ obtenues pour deux courbes identiques à l'homothétie de rapport K près, donc nous avons $\{T_i^{(2)} = T_i^{(1)}, i = 0, ..., n\}$. Pour montrer que les courbes résultats $C^{(1)}$ et $C^{(2)}$ sont identiques à l'homothétie de rapport K près, il nous suffit de montrer que les deux polygones de contrôle associés aux courbes dérivées $\{d_i^{(1)}, i = 0, ..., n\}$ et $\{d_i^{(2)}, i = 0, ..., n\}$ sont égaux. En effet, si nous supposons cela vrai, alors nous avons par la définition de la spline cubique sur la sphère :

$$T^{(2)}(\sigma) = T^{(2)}(K.s) = T^{(1)}(s) \tag{1.47}$$

Et alors

$$\begin{aligned} C^{(2)}(\sigma) &= \int_0^{\sigma} T^{(2)}(t)dt = \int_0^{s} T^{(2)}(K.v)K.dv \\ &= K.\int_0^{s} T^{(1)}(v)dv = K.C^{(1)}(s) \end{aligned}$$

Ce qui montre l'invariance par homothétie.

Donc montrons l'égalité des polygones de contrôle. Ceci doit se faire en deux étapes : l'égalité lors de l'initialisation et lors de la récurrence.

Pour l'initialisation, nous utilisons une B-spline cubique, le premier polygone de contrôle est obtenu par inversion d'une matrice qui est identique pour les 2 ensembles de données (en effet dans la matrice (1.42), seuls interviennent des rapports de distances inter-capteurs) : nous obtenons alors le même polygone initial. Lors de la récurrence, le paramétrage n'intervient plus du tout, donc nous obtiendrons également la même suite de polygones lors de l'algorithme.

Nous avons ainsi prouvé le fait que cette méthode est invariante par homothétie.

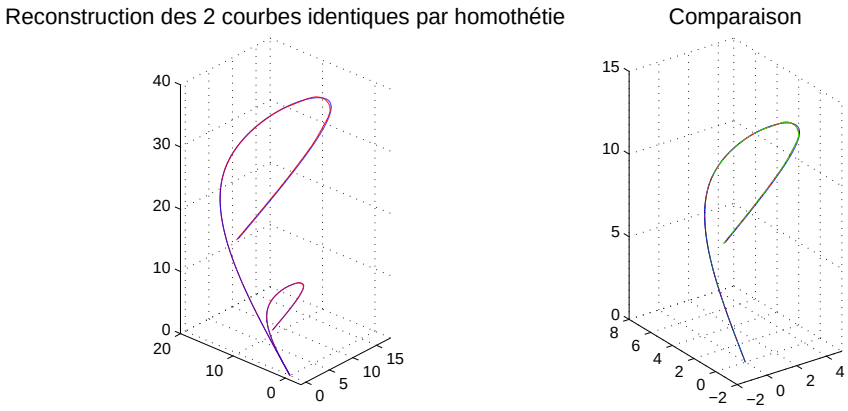


FIG. 1.33 – Invariance par homothétie

Nous pouvons en voir une illustration dans la figure FIG.1.33.

Nous voyons sur la gauche les deux courbes initiales en bleu et les deux courbes reconstruites en rouge, et la comparaison sur le graphe de droite où les deux courbes reconstruites sont confondues (rouge et vert).

1.2.6 CONVERGENCE

Nous allons maintenant étudier les propriétes de convergence de cette méthode de reconstruction. Tout comme pour les courbes planes, voyons ceci en deux étapes : la première consiste à majorer les erreurs de reconstruction de la courbe en fonction des errreurs de reconstruction de la courbe dérivée, puis nous nous intéresserons à l'erreur de reconstruction de la courbe dérivée sur la sphère.

Nous adoptons les notations suivantes : $C = (C_x, C_y, C_z)$ étant la courbe initiale et $\widetilde{C} = (\widetilde{C}_x, \widetilde{C}_y, \widetilde{C}_z)$ la courbe reconstruite.

A. CONVERGENCE DE LA COURBE SELON LA CONVERGENCE DE LA COURBE DÉRIVÉE

En norme infinie

$$\begin{aligned} \left\|C - \widetilde{C}\right\|_\infty &= \max\left(\left\|C_x - \widetilde{C}_x\right\|_\infty, \left\|C_y - \widetilde{C}_y\right\|_\infty, \left\|C_z - \widetilde{C}_z\right\|_\infty\right) \\ \left\|C_x - \widetilde{C}_x\right\|_\infty &= \sup_{s\in[0,L]} \left|C_x(s) - \widetilde{C}_x(s)\right| \end{aligned}$$

Or :

$$\begin{aligned} \left|C_x(s) - \widetilde{C}_x(s)\right| &= \left|\int_0^s C'_x(t) - \widetilde{C'}_x(t)\right| && (1.48a) \\ &\leq \sqrt{L}\left\|C'_x - \widetilde{C'}_x\right\|_2 \text{ par l'inégalité de Cauchy-Schwartz} && (1.48b) \end{aligned}$$

Donc :

$$\begin{aligned}\left\|C_x - \widetilde{C}_x\right\|_\infty &\leq \sqrt{L}\left\|C'_x - \widetilde{C'}_x\right\|_2 \\ \left\|C - \widetilde{C}\right\|_\infty &\leq \sqrt{L}\max\left(\left\|C'_x - \widetilde{C'}_x\right\|_2, \left\|C'_y - \widetilde{C'}_y\right\|_2, \left\|C'_z - \widetilde{C'}_z\right\|_2\right)\end{aligned}$$

Or

$$\left\|C'_x - \widetilde{C'}_x\right\|_2 \leq \left\|C' - \widetilde{C'}\right\|_2 \tag{1.49}$$

nous avons ainsi la majoration suivante :

$$\left\|C - \widetilde{C}\right\|_\infty \leq \sqrt{L}\left\|C' - \widetilde{C'}\right\|_2 \tag{1.50}$$

En norme 2

$$\begin{aligned}\left\|C - \widetilde{C}\right\|_2 &= \sqrt{\int_0^L \left\|C(s) - \widetilde{C}(s)\right\|_2^2 ds} \\ \left\|C(s) - \widetilde{C}(s)\right\|_2^2 &= \left(C_x(s) - \widetilde{C}_x(s)\right)^2 + \left(C_y(s) - \widetilde{C}_y(s)\right)^2 + \left(C_z(s) - \widetilde{C}_z(s)\right)^2\end{aligned}$$

Or d'après (1.48b)

$$\begin{aligned}\left(C_x(s) - \widetilde{C}_x(s)\right)^2 &\leq L\left\|C'_x - \widetilde{C'}_x\right\|_2^2 \\ \int_0^L \left(C_x(s) - \widetilde{C}_x(s)\right)^2 ds &\leq L^2\left\|C'_x - \widetilde{C'}_x\right\|_2^2\end{aligned}$$

$$\left\|C - \widetilde{C}\right\|_2 \leq L\sqrt{\left\|C'_x - \widetilde{C'}_x\right\|_2^2 + \left\|C'_y - \widetilde{C'}_y\right\|_2^2 + \left\|C'_z - \widetilde{C'}_z\right\|_2^2}$$

Donc en utilisant (1.49), nous obtenons la majoration finale :

$$\left\|C - \widetilde{C}\right\|_2 \leq \sqrt{3}L\left\|C' - \widetilde{C'}\right\|_2 \tag{1.51}$$

En norme 1 Par l'inégalité de Cauchy-Schwartz, nous avons tout de suite une majoration de cette erreur :

$$\left\|C - \widetilde{C}\right\|_1 \leq L\sqrt{3L}\left\|C' - \widetilde{C'}\right\|_2 \tag{1.52}$$

B. CONVERGENCE DE LA COURBE DÉRIVÉE

D'après les calculs qui précèdent, nous voyons que si nous voulons montrer la convergence de la méthode élaborée, nous devons montrer la convergence en norme $L^{(2)}$ de la courbe dérivée :

$$\begin{aligned}\left\|C' - \widetilde{C'}\right\|_2 &= \sqrt{\int_0^L \left\|C'(s) - \widetilde{C'}(s)\right\|_2^2 ds} \\ \left\|C'(s) - \widetilde{C'}(s)\right\|_2^2 &= \left(C'_x(s) - \widetilde{C'}_x(s)\right)^2 + \left(C'_y(s) - \widetilde{C'}_y(s)\right)^2 + \left(C'_z(s) - \widetilde{C'}_z(s)\right)^2\end{aligned}$$

La courbe dérivée a été obtenue par une spline sur la sphère. Ces courbes ayant été assez peu étudiées, aucun résultat de convergence n'a été prouvé à notre connaissance. Cependant, en voyant le comportement des erreurs en fonction du nombre de capteurs (dans la partie qui suit) et du fait que ces courbes sont obtenues par analogie des splines dans le plan qui ont, elles, des propriétés de convergence, nous pouvons supposer qu'une telle propriété est envisageable.

Ainsi, si nous supposons avoir une convergence des splines sur la sphère, nous aurons alors convergence de la méthode de reconstruction. Afin de juger du caractère convergent de cette méthode, nous allons par la suite étudier l'évolution des erreurs en fonction du nombre de capteurs sur quelques exemples.

1.2.7 COURBES D'ERREUR EN FONCTION DE CERTAINS PARAMÈTRES

A. ERREURS EN FONCTION DU NOMBRE DE CAPTEURS

Test 1 : spline 3D Voici la courbe que nous testons (voir FIG.1.34). Nous avons appliqué

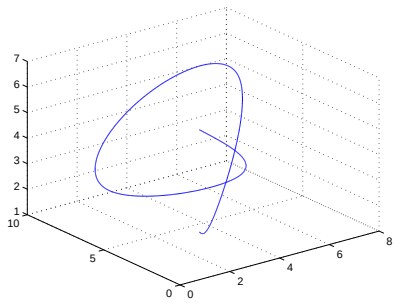
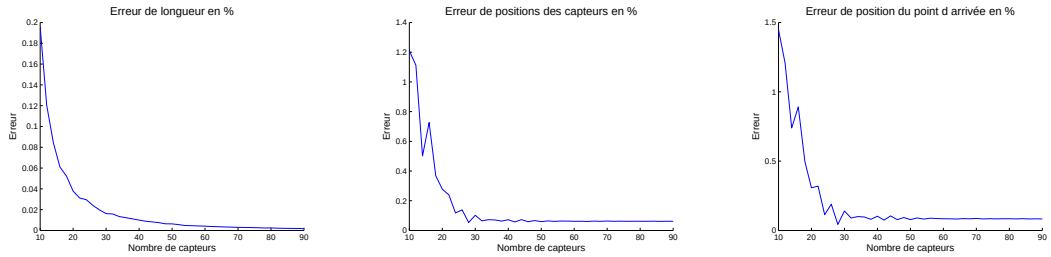
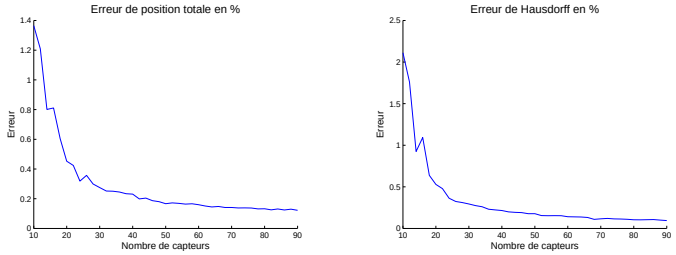


FIG. 1.34 – Spline 3D

notre méthode de reconstruction pour un nombre de capteurs variant de 10 à 90 avec un pas de 2. Voici les différentes courbes d'erreur obtenues (voir FIG.1.35) :



(a) Erreur de longueur (b) Erreur de position des capteurs (c) Erreur de position du dernier capteur



(d) Erreur de position totale (e) Erreur de Hausdorff

FIG. 1.35 – Erreurs pour la spline 3D

Test 2 : courbe fermée Voici la courbe que nous testons (voir FIG.1.36) : La figure

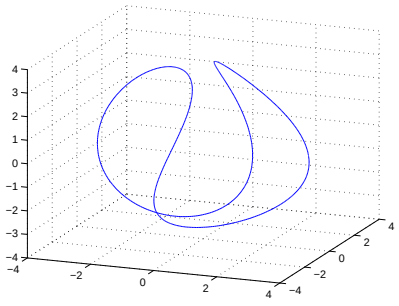
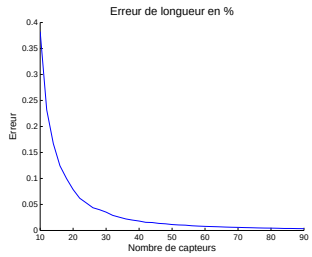
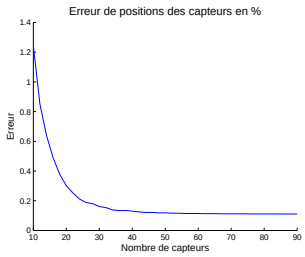


FIG. 1.36 – Courbe fermée

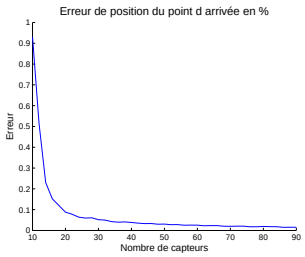
FIG.1.37 regroupe les erreurs pour le même ensemble de tests pour que la première courbe.



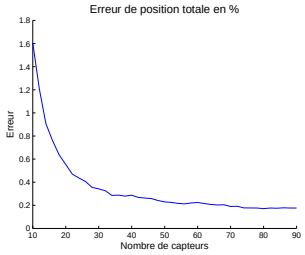
(a) Erreur de longueur



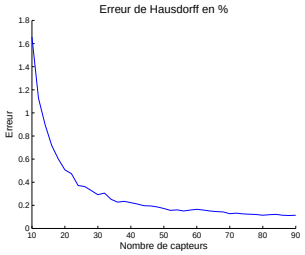
(b) Erreur de position des capteurs



(c) Erreur de position du dernier capteur



(d) Erreur de position totale



(e) Erreur de Hausdorff

FIG. 1.37 – Erreurs pour la courbe fermée

Test 3 : courbe hélicoïdale à rayon variable La courbe FIG.1.38 représente la courbe testée. Les résultats de tests pour un nombre de capteurs variant de 20 à 90 avec un pas

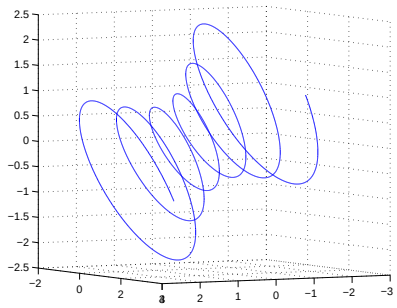
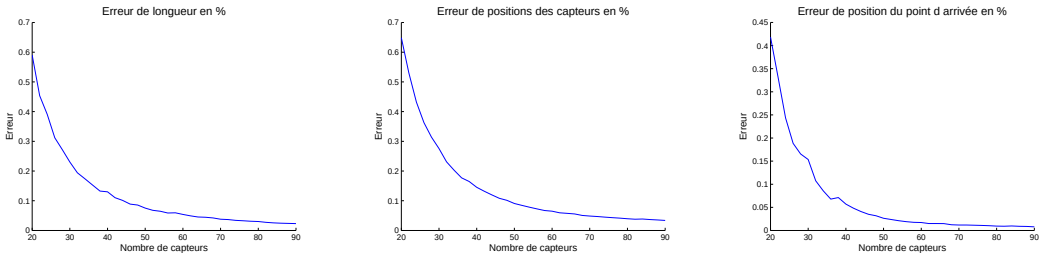
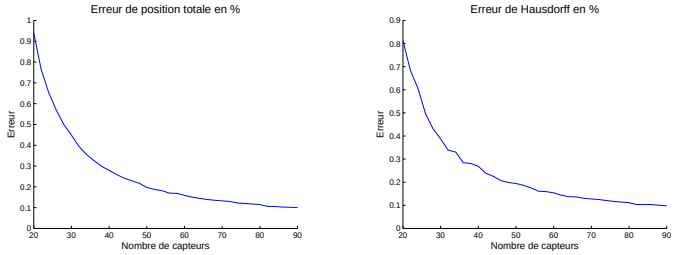


FIG. 1.38 – Courbe hélicoïdale à rayon variable

de 2 sont rassemblés dans la figure FIG.1.39.



(a) Erreur de longueur (b) Erreur de position des capteurs (c) Erreur de position du dernier capteur



(d) Erreur de position totale (e) Erreur de Hausdorff

FIG. 1.39 – Erreurs pour la courbe hélicoïdale à rayon variable

Commentaires Nous obtenons sur ces différents exemples une rapide décroissance des erreurs en fonction du nombre de capteurs, qui nous permet de dire que nous pouvons obtenir de très bonnes reconstructions à partir d'un nombre assez faible de capteurs et nous laisse penser que notre méthode satisfait les critères de convergence.

B. QUELQUES TESTS AVEC DES DONNÉES BRUITÉES

Tout comme pour les courbes planes, nous allons voir ici quelques exemples de données bruitées, soit au niveau des valeurs des capteurs, soit au niveau de leur positionnement.

B.a. Bruiter les valeurs des capteurs Nous allons ici bruiter les valeurs des capteurs. Ces capteurs nous donnent des vecteurs tangents unitaires. Afin de bruiter les valeurs de ces vecteurs, (afin que ces erreurs de valeurs aient un sens par rapport à la réalité), nous allons décomposer ces vecteurs en paramétrisation sphérique, et nous allons bruiter les deux angles de cette paramétrisation sphérique avec un bruit gaussien de moyenne nulle et de variance variable (nous pourrons alors dire que les vecteurs sont perturbés dans un domaine "rectangulaire" sur la sphère unité). Les données seront connues à $\pm k°$, cela signifie que les deux angles seront bruités avec une variance de $\sigma = k.\frac{\pi}{3.180}$. Voyons ce que nous obtenons pour certaines courbes tests. Chaque point de mesure d'erreur est obtenu en moyennant 10 erreurs sur 10 courbes bruitées de même variance. Nous calculons ici seulement les erreurs de position globale.

Test 1 : spline 3D Voyons tout d'abord un résultat de courbe obtenu avec 20 capteurs et des valeurs de capteurs connues à $\pm 10°$ près (voir FIG.1.40) : l'erreur de reconstruction obtenue est de 0.97%. Nous voyons en bleu la courbe initiale, en rouge la

reconstruction non bruitée, et en vert la reconstruction utilisant les données bruitées.

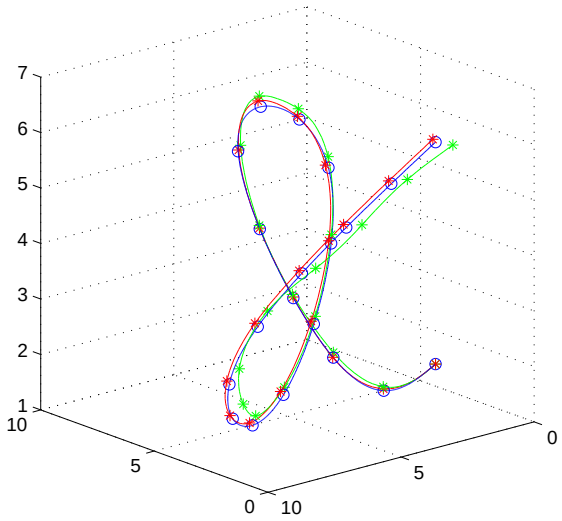


FIG. 1.40 – Reconstruction bruitée pour la spline 3D,*(bleu) : courbe initiale, (rouge) : courbe reconstruite à partir de données non bruitées, (vert) : courbe reconstruite à partir de données bruitées*

Voici les résultats que nous obtenons pour la même batterie de tests que pour les courbes planes : nous faisons varier l'amplitude du bruit pour 15, 20 et 25 capteurs pour les bruits suivants : $\{1°, 2°, 4°, 6°, 8°, 10°, 15°\}$. Il y a donc 3 courbes d'erreurs pour des bruits allant de $\pm 1°$ à $\pm 15°$ (voir FIG.1.41).

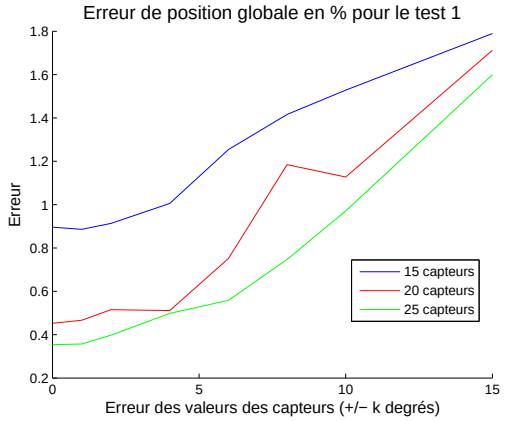


FIG. 1.41 – Erreur en fonction des erreurs des valeurs capteurs pour la spline 3D

Test 2 : courbe fermée Voyons ici le deuxième exemple, avec tout d'abord un aperçu visuel d'un résultat obtenu avec 20 capteurs et des données bruitées à $\pm 10°$ près (voir FIG.1.42), l'erreur de reconstruction est de 1.33%.

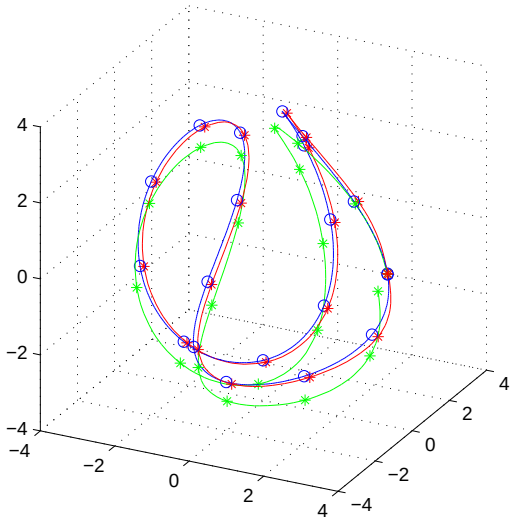


FIG. 1.42 – Reconstruction bruitée pour la courbe fermée, *(bleu) : courbe initiale, (rouge) : courbe reconstruite à partir de données non bruitées, (vert) : courbe reconstruite à partir de données bruitées*

Nous effectuons la même batterie de tests, pour 15, 20 et 25 capteurs, les résultats sont visibles sur la figure FIG.1.43.

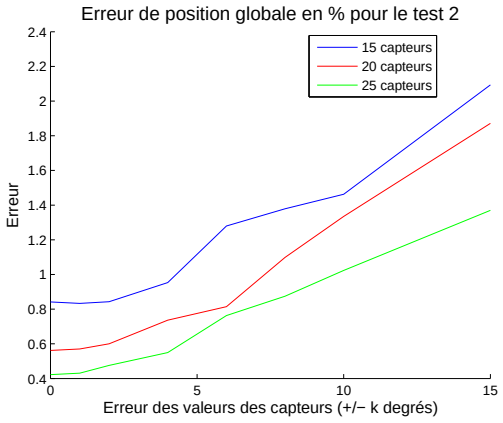


FIG. 1.43 – Erreur en fonction des erreurs des valeurs capteurs pour la courbe fermée

Commentaires Nous voyons sur ces exemples que nous obtenons des courbes lisses (le fait que les données soient bruitées ne perturbent pas l'aspect lisse de la courbe). Les

erreurs croissent assez faiblement en fonction de la variance de l'erreur, et les erreurs décroissent assez vite en fonction du nombre de capteurs. Nous en déduisons ainsi que la méthode est assez robuste vis-à-vis du bruit, et que le nombre de capteurs compense leurs erreurs de précision.

B.b. Bruiter les positions des capteurs Nous allons ici bruiter la position des capteurs de la même façon que pour les courbes planes.

Test 1 : spline 3D Voyons tout d'abord un résultat de courbe obtenu avec 20 capteurs et des positions de capteurs connues à $\pm 10\%$ près (voir FIG.1.44) : l'erreur de reconstruction obtenue est de 1.14%.

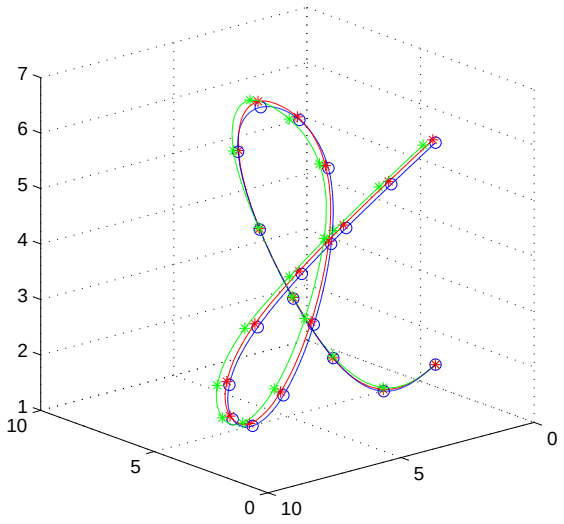


FIG. 1.44 – Reconstruction bruitée pour la spline 3D, *(bleu) : courbe initiale, (rouge) : courbe reconstruite à partir de données non bruitées, (vert) : courbe reconstruite à partir de données bruitées*

Nous faisons les mêmes tests que pour les courbes planes : nous faisons varier l'amplitude du bruit des positions pour 6, 10 et 15 capteurs pour les bruits de positions suivants : $\{1\%, 2\%, 4\%, 6\%, 8\%, 10\%, 15\%\}$. Il y a donc 3 courbes d'erreurs pour des bruits allant de $\pm 1\%$ à $\pm 15\%$ (voir FIG.1.45).

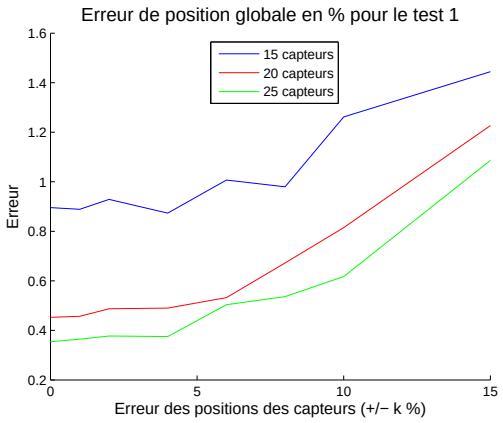


FIG. 1.45 – Erreur en fonction des erreurs des positions des capteurs pour la spline 3D

Test 2 : courbe fermée Voyons dans un premier temps un résultat obtenu avec 20 capteurs et des positions de capteurs bruitées à $\pm 10\%$ près sur la figure FIG.1.46 ; l'erreur de reconstruction obtenue est de 0.88%.

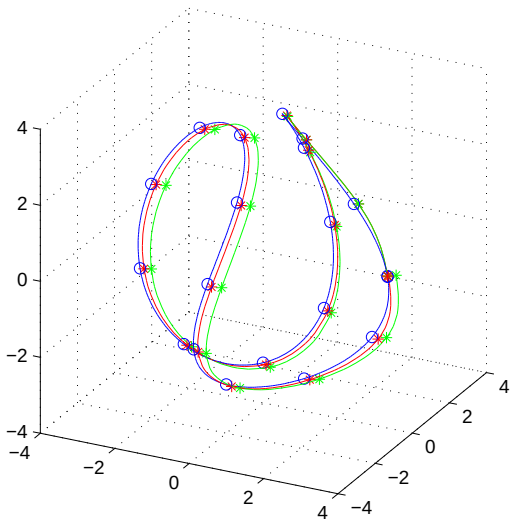


FIG. 1.46 – Reconstruction bruitée pour la courbe fermée, *(bleu) : courbe initiale, (rouge) : courbe reconstruite à partir de données non bruitées, (vert) : courbe reconstruite à partir de données bruitées*

Les erreurs sont également calculées dans les mêmes conditions que pour le test précédent et sont situées sur la figure FIG.1.47.

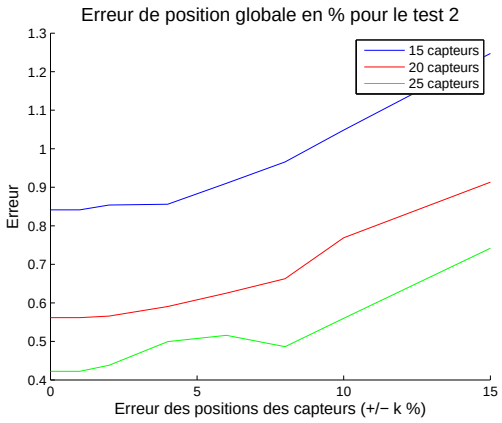


FIG. 1.47 – Erreur en fonction des erreurs des positions des capteurs pour le test 2

Commentaires Nous voyons d'après ces résultats que nous obtenons des erreurs assez faibles, même pour des erreurs assez importantes (beaucoup plus importantes que ce qui peut arriver dans des cas réels). Donc nous obtiendrons de bonnes reconstructions avec cette méthode. Nous voyons également sur ces exemples que pour diminuer l'erreur, il nous faudra augmenter le nombre de capteurs (le nombre d'informations compense la qualité de celle-ci).

1.2.8 CONCLUSION

En conclusion de cette partie, nous pouvons dire que nous avons pu étendre la méthode mise en place pour les courbes planes, c'est-à-dire reconstruire tout d'abord la dérivée de la fonction paramétrée par l'abscisse curviligne, puis intégrer cette fonction obtenue par une méthode numérique. La reconstruction de la dérivée a fait appel à des reconstructions de courbes sur la sphère. Nous avons prouvé des propriétés d'invariance (par rotation et homothétie), un sens physique (minimisation d'une quantité alliant courbure, variation de courbure et torsion qui sont des données intrinsèques à la courbe). Nous avons également démontré des propriétés de convergence, et étudié quelques courbes d'erreur en fonction du nombre de capteurs, et des erreurs possibles données par les capteurs, afin de voir la robustesse de notre méthode.

Dans la suite de ce document, nous allons insérer des connaissances a priori sur les positions de la courbe. En effet, une des perspectives de ce travail est le suivi dans le temps de formes géométriques. Deux choix s'ouvrent à nous : soit à chaque pas de temps nous utilisons la méthode que nous venons de voir ; soit nous utilisons l'information de position de la courbe au temps précédent. C'est cette dernière solution que nous allons étudier dans la prochaine section.

Chapitre 2

Déformation de courbes

Ce chapitre apporte une nouvelle vision au problème. Nous étudions ici le problème de suivi de déformation d'une courbe dans le temps. Nous devons alors répondre à la même problématique que précédemment, à savoir reconstruire une courbe qui admet des contraintes de tangente en certains points de répartition curviligne connue, mais nous connaissons ici la position de la courbe au temps précédent.
Ce suivi de déformation peut être un vrai suivi de déformation, c'est-à-dire que les données au cours du temps sont vraiment les données d'une courbe se déformant au cours du temps, mais cela peut également être une alternative pour reconstruire une courbe statique (rappelant les homotopies), où dans ce cas les données générées entre la courbe de départ (étant prise par exemple comme un segment de droite) et la courbe finale (celle que nous cherchons à reconstruire) sont seulement des données intermédiaires permettant l'évolution lente de la solution.
La méthode que nous élaborons ici est une méthode directe : nous modélisons la courbe dans l'espace de $\mathbb{R}^3$ directement sans passer par l'espace tangentiel (ce que nous faisions dans le chapitre précédent). Nous utilisons la décomposition dans la base de Hermite (base des polynômes de degré inférieur à 3), ce qui nous permet de contrôler directement dans le modèle les positions des capteurs et leurs données tangentielles. La résolution du problème se fera alors par les contraintes de longueur à respecter entre chaque capteur.
La validation de cette méthode se fera par l'étude des erreurs sur des exemples, et par comparaison avec la méthode validée dans le chapitre précédent.
Les méthodes développées dans cette partie seront également largement réutilisées dans la suite pour la reconstruction des surfaces, mais également pour leur déformation.

Sommaire

Une des perspectives de ce travail est le suivi de déformation. Nous avons ainsi des connaissances supplémentaires pour la reconstruction d'une courbe en un temps donné, c'est-à-dire qur nous connaissons la position de cette courbe dans un temps précédent assez proche de la position inconnue actuelle. Ceci nous amène à étudier le nouveau problème suivant : nous supposons que nous connaissons la position de la courbe à un temps donné, et nous allons chercher à exprimer la forme de cette courbe au temps suivant en nous aidant de la position au temps précédent, et de ses nouvelles données tangentielles. Pour la suite de ce travail, nous supposons que les courbes que nous reconstruisons sont assez régulières pour définir les dérivées successives dont nous aurons besoin.

2.1 MÉTHODE

2.1.1 FORMALISATION

Nous supposons que nous avons une courbe au temps t qui répond aux contraintes de "reconstruction de courbes à partir des données tangentielles". Donc nous connaissons :

- la courbe $f(s)$ au temps t et $f_i := f|_{[s_i,s_{i+1}]}$,
- les tangentes au temps t : $T_i, i = 0, ..., n$, aux positions curvilignes des capteurs $s_i, i = 0, ..., n$,
- les positions des capteurs au temps t : $P_i, i = 0, ..., n$.

Nous voulons modifier cette courbe afin qu'elle réponde aux contraintes au temps $t+\delta t$. Les nouvelles données sont des nouvelles valeurs de tangentes aux positions des capteurs $\widetilde{T_i}, i = 0, ..., n$.

Nous proposons ici de modéliser la nouvelle courbe $\widetilde{f}$ de sorte qu'il soit immédiat de contrôler les positions des capteurs et les tangentes en ces capteurs : nous allons donc moduler par morceaux l'ancienne courbe f avec les fonctions de Hermite φ, qui forment une base de l'ensemble des fonctions polynomiales de degré inférieur ou égal à 3. Cette modélisation va nous permettre de définir la courbe seulement à partir de ces points de positions des capteurs (P_i à l'instant précédent et $\widetilde{P_i}$ à l'instant présent) et des valeurs des tangentes (T_i à l'instant précédent et $\widetilde{T_i}$ à l'instant présent) :

$$\begin{aligned}\widetilde{f_k(s)} &= f_k(s) + D_k^P \varphi_0\left(\frac{s-s_k}{h_k}\right) + D_{k+1}^P \varphi_1\left(\frac{s-s_k}{h_k}\right) \\ &+ h_k.D_k^T \varphi_2\left(\frac{s-s_k}{h_k}\right) + h_k.D_{k+1}^T \varphi_3\left(\frac{s-s_k}{h_k}\right), k = 0, ..., n-1 \qquad (2.1)\end{aligned}$$

pour s tel que $s_k \leq s \leq s_{k+1}$, avec :

$$\begin{aligned} h_k &= s_{k+1} - s_k, k = 0, ..., n-1 \\ u &= \frac{s - s_k}{h_k} \\ \varphi_0(u) &= 2u^3 - 3u^2 + 1 \\ \varphi_1(u) &= -2u^3 + 3u^2 \\ \varphi_2(u) &= u(u-1)^2 \\ \varphi_3(u) &= u^2(u-1) \end{aligned}$$

où les fonctions φ sont les fonctions de Hermite (voir FIG.2.1), et où nous avons défini les

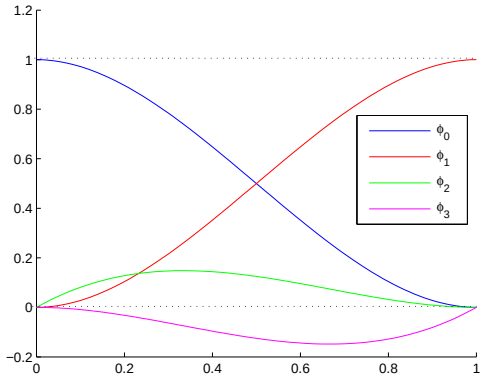


FIG. 2.1 – Les 4 fonctions de Hermite

différences de valeurs de positions et tangentes de la façon suivante :

$$\begin{aligned} D_k^P &= \widetilde{P}_k - P_k \\ D_k^T &= \widetilde{T}_k - T_k \end{aligned}$$

La figure FIG.2.2 illustre la modélisation que nous avons choisie.

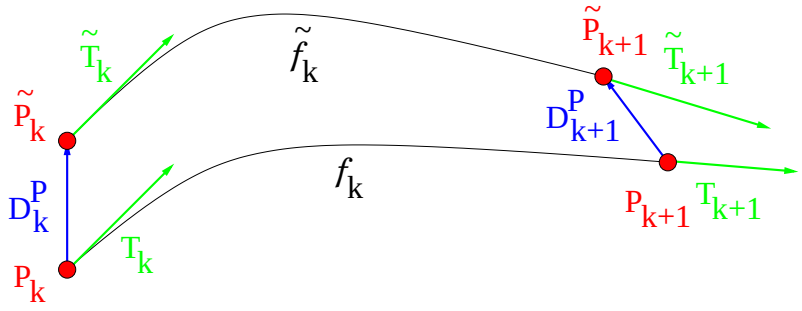


FIG. 2.2 – Courbe initiale et courbe déformée

Les inconnues sont alors les $\widetilde{P}_k$, $k = 1, ..., n$, qui sont les nouvelles positions des capteurs (nous supposons que le premier capteur ne bouge pas (car point de départ relatif)).

Cette modélisation satisfait les contraintes de positions et tangentes aux extrémités :

$$\widetilde{f_k(s_k)} = f_k(s_k) + D_k^P = P_k + \widetilde{P}_k - P_k = \widetilde{P}_k \quad (2.2a)$$

$$\widetilde{f_k(s_{k+1})} = f_k(s_{k+1}) + D_{k+1}^P = P_{k+1} + \widetilde{P}_{k+1} - P_{k+1} = \widetilde{P}_{k+1} \quad (2.2b)$$

$$\widetilde{f_k'(s_k)} = f_k'(s_k) + D_k^T = T_k + \widetilde{T}_k - T_k = \widetilde{T}_k \quad (2.2c)$$

$$\widetilde{f_k'(s_{k+1})} = f_k'(s_{k+1}) + D_{k+1}^T = T_{k+1} + \widetilde{T}_{k+1} - T_{k+1} = \widetilde{T}_{k+1} \quad (2.2d)$$

Nous devons ainsi déterminer les inconnues à partir de la seule contrainte non encore satisfaite dans le modèle : la conservation de longueur. Les contraintes de longueur entre chaque capteur sont de la forme suivante :

$$h_k = \int_{s_k}^{s_{k+1}} \left\| \widetilde{f_k'(t)} \right\| dt, k = 0, ..., n-1. \quad (2.3)$$

Nous avons ainsi n points inconnus pour n contraintes. La première contrainte ne fait intervenir qu'une seule inconnue : D_1^P, les autres contraintes en font intervenir deux : D_k^P et D_{k+1}^P. Nous avons alors décidé de découpler les équations, ainsi, en satisfaisant les contraintes les unes après les autres, la contrainte de longueur h_k a pour seule inconnue le point d'arrivée du segment : D_{k+1}^P.

Note : Nous pourrions avoir une vision plus globale en résolvant le problème sans découpler les équations, cela permettrait alors de traiter d'autres cas de figure en fixant les contraintes supplémentaires voulues (par exemple fixer certains points de passage ou fermer la courbe....).

Le problème revient ici à résoudre n problèmes identiques successivement : nous cherchons le point d'arrivée D_{k+1}^P du segment $\widetilde{f}_k$ en ayant pour contrainte la longueur de ce segment de courbe h_k.

2.1.2 Résolution

Pour le calcul de la contrainte de la longueur h_k, nous devons calculer $\left\| \widetilde{f}_k{}' \right\|$.

Nous calculons $\left\| \widetilde{f_k'(t)} \right\|$ par produit scalaire :

$$\begin{aligned} \left\| \widetilde{f}_k' \right\|^2 &= \left\langle \widetilde{f}_k', \widetilde{f}_k' \right\rangle \\ &= \left\langle \left(f_k' + D_k^P \frac{\varphi_0'}{h_k} + D_{k+1}^P \frac{\varphi_1'}{h_k} + D_k^T \varphi_2' + D_{k+1}^T \varphi_3' \right) \right. \\ &\quad \left. , \left(f_k' + D_k^P \frac{\varphi_0'}{h_k} + D_{k+1}^P \frac{\varphi_1'}{h_k} + D_k^T \varphi_2' + D_{k+1}^T \varphi_3' \right) \right\rangle \end{aligned}$$

Lors du calcul de la contrainte de longueur h_k, la seule inconnue est le point d'arrivée, $\widetilde{P}_{k+1}$, c'est à dire le déplacement D_{k+1}^P.

Si l'on sépare ce qui est connu de ce qui ne l'est pas dans $\widetilde{f_k}$, nous obtenons :

$$\widetilde{f_k} = \left(f_k + D_k^P\varphi_0 + h_k D_k^T\varphi_2 + h_k D_{k+1}^T\varphi_3\right) + D_{k+1}^P\varphi_1$$

Ainsi, lors du calcul de la norme de $\widetilde{f_k}'$, nous obtenons :

$$\begin{aligned}\left\langle \widetilde{f_k'}, \widetilde{f_k'} \right\rangle &= \left\langle \left(f_k' + D_k^P\frac{\varphi_0'}{h_k} + D_k^T\varphi_2' + D_{k+1}^T\varphi_3'\right), \left(f_k' + D_k^P\frac{\varphi_0'}{h_k} + D_k^T\varphi_2' + D_{k+1}^T\varphi_3'\right)\right\rangle \\ &+2\left\langle \left(f_k' + D_k^P\frac{\varphi_0'}{h_k} + D_k^T\varphi_2' + D_{k+1}^T\varphi_3'\right), D_{k+1}^P\frac{\varphi_1'}{h_k}\right\rangle \\ &+\left\langle D_{k+1}^P\frac{\varphi_1'}{h_k}, D_{k+1}^P\frac{\varphi_1'}{h_k}\right\rangle\end{aligned}$$

Si l'on pose les inconnues scalaires x, y, z tels que $D_{k+1}^P = (x, y, z)$, alors nous avons :

$$\left\langle \widetilde{f_k'}(s), \widetilde{f_k'}(s)\right\rangle = a(t) + b_1(t)x + b_2(t)y + b_3(t)z + \frac{\varphi_1'^2(t)}{h_k^2}(x^2+y^2+z^2) \tag{2.4}$$

avec

$$\begin{aligned} t &= \frac{s-s_k}{h_k},\ t\in[0,1] \\ a(t) &= \left\langle \left(f_k'(t) + D_k^P\frac{\varphi_0'(t)}{h_k} + D_k^T\varphi_2'(t) + D_{k+1}^T\varphi_3'(t)\right), \right. \\ &\quad \left. \left(f_k'(t) + D_k^P\frac{\varphi_0'(t)}{h_k} + D_k^T\varphi_2'(t) + D_{k+1}^T\varphi_3'(t)\right)\right\rangle \\ b_i(t) &= 2\frac{\varphi_1'(t)}{h_k}\left(f_{k,i}'(t) + D_{k,i}^P.\frac{\varphi_0'(t)}{h_k} + D_{k,i}^T.\varphi_2'(t) + D_{k+1,i}^T.\varphi_3'(t)\right) \\ &\quad i=1,2,3 \text{ où } D_{k,i}^P \text{ est la } i\text{-eme coordonnée de } D_k^P, etc... \end{aligned} \tag{2.5}$$

Nous avons alors la contrainte :

$$\begin{aligned} h_k &= \int_{s_k}^{s_{k+1}} \left\|\widetilde{f_k'(s)}\right\| ds, k=0,...,n-1 \\ h_k &= \int_{s_k}^{s_{k+1}} \sqrt{a(t) + b_1(t)x + b_2(t)y + b_3(t)z + \frac{\varphi_1'^2(t)}{h_k^2}(x^2+y^2+z^2)}\ ds \end{aligned}$$

et par changement de variables, nous obtenons

$$1 = \int_0^1 \sqrt{a(t) + b_1(t)x + b_2(t)y + b_3(t)z + \frac{\varphi_1'^2(t)}{h_k^2}(x^2+y^2+z^2)}\ dt \tag{2.6}$$

Nous devons alors intégrer la racine d'une fonction, qui est une forme quadratique en les inconnues.

Nous allons calculer cette intégrale avec une méthode numérique, la méthode de Simpson avec N pas d'échantillonnage :

$$1 = \frac{1}{3N}\left(f_1 + f_{N+1} + 4\sum_{\substack{i=2\\ i \text{ pair}}}^{N} f_i + 2\sum_{\substack{i=3\\ i \text{ impair}}}^{N} f_i\right)$$

avec

$$f_i = \sqrt{a\left(\frac{i-1}{N}\right) + b_1\left(\frac{i-1}{N}\right)x + b_2\left(\frac{i-1}{N}\right)y + b_3\left(\frac{i-1}{N}\right)z + \frac{\varphi_1'^2\left(\frac{i-1}{N}\right)}{h_k^2}(x^2+y^2+z^2)}$$

Notre contrainte apparait maintenant comme une somme finie de racines carrées où les 3 inconnues apparaissent sous forme quadratique à l'intérieur de chaque racine carrée.

Cette équation n'ayant pas une solution unique, nous devons choisir la meilleure solution. Nous allons donc prendre la solution qui minimise l'énergie de la courbe :

$$\begin{aligned}\min E &= \min \int \left\|\widetilde{f}''\right\|^2 \\ &= \min \sum_{k=0}^{n-1} h_k \int_0^1 \left\|\widetilde{f_k}''(t)\right\|^2 dt \end{aligned} \tag{2.7}$$

Comme nous avons découplé les équations pour trouver les inconnues, nous allons également minimiser indépendamment les énergies sur chaque morceau : nous avons alors une contrainte de minimisation associée à une contrainte d'égalité de longueur sur chaque morceau (nous pouvons ainsi tout résoudre de manière successive). La minimisation revient à minimiser une fonction quadratique en les inconnues (x, y, z) ; les calculs sont similaires à ceux du calcul de la longueur :

$$\left\|\widetilde{f_k}''(s)\right\|^2 = \alpha(t) + \beta_1(t)x + \beta_2(t)y + \beta_3(t)z + \frac{\varphi_1''^2(t)}{h_k^4}(x^2+y^2+z^2) \tag{2.8}$$

avec

$$\begin{aligned}\alpha(t) &= \left\langle \left(f_k''(t) + D_k^P\frac{\varphi_0''(t)}{h_k^2} + D_k^T\frac{\varphi_2''(t)}{h_k} + D_{k+1}^T\frac{\varphi_3''(t)}{h_k}\right), \right. \\ &\quad \left. \left(f_k''(t) + D_k^P\frac{\varphi_0''(t)}{h_k^2} + D_k^T\frac{\varphi_2''(t)}{h_k} + D_{k+1}^T\frac{\varphi_3''(t)}{h_k}\right)\right\rangle \\ \beta_i(t) &= 2\frac{\varphi_1''(t)}{h_k}\left(f_{k,i}''(t) + D_{k,i}^P.\frac{\varphi_0''(t)}{h_k} + D_{k,i}^T.\varphi_2''(t) + D_{k+1,i}^T.\varphi_3''(t)\right) \\ &\quad i = 1,2,3 \text{ où } D_{k,i}^P \text{ est la } i\text{-eme coordonnée de } D_k^P, etc...\end{aligned} \tag{2.9}$$

Notre contrainte de minimisation est une contrainte quadratique en les inconnues :

$$\min\left(I_\alpha + I_{\beta_1}x + I_{\beta_2}y + I_{\beta_3}z + I_{\varphi_1''^2}(x^2+y^2+z^2)\right) \tag{2.10}$$

$$\text{avec } I_\alpha = h_k\int_0^1 \alpha(t)\,dt, etc...$$

2.1.3 UTILISATION

Afin de simplifier les calculs, nous supposons qu'à l'étape initiale, nous avons une fonction composée de fonctions de Hermite. Ainsi après chaque itération du calcul, nous obtenons des fonctions qui sont des combinaisons linéaires des fonctions de Hermite, et la connaissance seule des positions des capteurs et de leur information tangentielle suffit à connaître exactement la formulation explicite de la fonction.

Ainsi, la contrainte de longueur entre deux capteurs s'écrit toujours comme une somme finie de racines carrées où les inconnues sont quadratiques sous les racines (cette contrainte a une formulation assez simple, dans le sens où nous pouvons facilement en exprimer le gradient, utile pour la résolution), et lors de la minimisation de l'énergie, les fonctions α et β_i sont des fonctions polynomiales, et nous connaissons alors exactement les intégrales I_α et I_{β_i}.

En résumé, notre méthode revient à chercher le triplet (x, y, z) tel que :

$$\left\{ \begin{array}{c} \min\left(I_\alpha + I_{\beta_1}x + I_{\beta_2}y + I_{\beta_3}z + I_{\varphi_1'^2}(x^2+y^2+z^2)\right) \\ 1 = S_{simp}\left(\sqrt{a + b_1x + b_2y + b_3z + \frac{\varphi_1'^2}{h_k^2}(x^2+y^2+z^2)}\right) \end{array} \right. \tag{2.11}$$

avec

$$\begin{array}{rcl} t & = & \dfrac{s - s_k}{h_k} \\ a(t) & = & \left\langle \left(\widetilde{P}_k\dfrac{\varphi_0'(t)}{h_k} + P_{k+1}\dfrac{\varphi_1'(t)}{h_k} + \widetilde{T}_k\varphi_2'(t) + \widetilde{T}_{k+1}\varphi_3'(t)\right), \right. \\ & & \left. \left(\widetilde{P}_k\dfrac{\varphi_0'(t)}{h_k} + P_{k+1}\dfrac{\varphi_1'(t)}{h_k} + \widetilde{T}_k\varphi_2'(t) + \widetilde{T}_{k+1}\varphi_3'(t)\right)\right\rangle \\ b_i(t) & = & 2\dfrac{\varphi_1'(t)}{h_k}\left(\widetilde{P}_{k,i}.\dfrac{\varphi_0'(t)}{h_k} + P_{k+1,i}.\dfrac{\varphi_1'(t)}{h_k} + \widetilde{T}_{k,i}.\varphi_2'(t) + \widetilde{T}_{k+1,i}.\varphi_3'(t)\right) \\ & & i = 1,2,3 \text{ où } \widetilde{P}_{k,i} \text{ est la } i\text{-eme coordonnée de } \widetilde{P}_k, etc \end{array} \tag{2.12}$$

et

$$\begin{array}{rcl} \alpha(t) & = & \left\langle \left(\widetilde{P}_k\dfrac{\varphi_0''(t)}{h_k^2} + P_{k+1}\dfrac{\varphi_1''(t)}{h_k^2} + \widetilde{T}_k\dfrac{\varphi_2''(t)}{h_k} + \widetilde{T}_{k+1}\dfrac{\varphi_3''(t)}{h_k}\right), \right. \\ & & \left. \left(\widetilde{P}_k\dfrac{\varphi_0''(t)}{h_k^2} + P_{k+1}\dfrac{\varphi_1''(t)}{h_k^2} + \widetilde{T}_k\dfrac{\varphi_2''(t)}{h_k} + \widetilde{T}_{k+1}\dfrac{\varphi_3''(t)}{h_k}\right)\right\rangle \\ \beta_i(t) & = & 2\dfrac{\varphi_1''(t)}{h_k^2}\left(\widetilde{P}_{k,i}.\dfrac{\varphi_0''(t)}{h_k^2} + P_{k+1,i}.\dfrac{\varphi_1''(t)}{h_k^2} + \widetilde{T}_{k,i}.\dfrac{\varphi_2''(t)}{h_k} + \widetilde{T}_{k+1,i}.\dfrac{\varphi_3''(t)}{h_k}\right) \\ & & i = 1,2,3 \text{ où } \widetilde{P}_{k,i} \text{ est la } i\text{-eme coordonnée de } \widetilde{P}_k, etc \end{array} \tag{2.13}$$

Ainsi, pour les calculs, nous avons seulement besoin de connaître les positions des capteurs au temps précédent, et les tangentes au temps présent (donc le fait de supposer que la courbe à l'état initial était une combinaison des courbes de Hermite nous a permis de

simplifier les calculs et ne pose finalement pas de problèmes car nous avons seulement besoin des données de positions et de tangentes aux points de mesure).

Lors de la minimisation, nous avons besoin d'un point initial. Pour le calcul de D^P_{k+1}, nous initialisons sa valeur à D^P_k, résultat trouvé lors du calcul sur le segment précédent (car nous supposons qu'assez naturellement la valeur du déplacement d'un point dépend du déplacement du point du segment précédent).

D'autres points sont paramétrables pour l'utilisation de cette méthode, comme le nombre de pas utilisés pour la méthode de Simpson, où la méthode de minimisation utilisée. Ces différents choix pourraient faire l'objet d'une étude théorique pour améliorer les résultats obtenus. Nous avons quant à nous fixé ces choix en fonction des premiers résultats obtenus, qui nous paraissaient satisfaisants.

2.2 TESTS

Nous allons dans cette partie comparer sur une série de tests de courbes simulées en déformation les deux méthodes que nous avons développées : la méthode qui consiste à reconstruire à chaque moment la courbe seulement à partir des données tangentielles, et celle qui construit la courbe en fonction de la position précédente et les nouvelles données capteurs.

Nous calculons les mêmes erreurs que dans la première partie (voir la partie 1.1.3 en page 32), à savoir l'erreur de longueur, l'erreur de positions des capteurs, l'erreur globale de position et l'erreur de Hausdorff.

2.2.1 TESTS COURBES PLANES

Afin de tester ces algorithmes de suivi de déformation, nous avons besoin de générer des courbes en déformation de longueur constante. Ainsi, nos courbes tests sont des courbes PH cubiques par morceaux, car le calcul de leur longueur est facile, et les déformer en contraignant leur longueur également (voir en annexe C).

Pour chaque test, nous allons tout d'abord voir la séquence de courbes simulées en déformation. Puis nous montrerons quelques résultats obtenus à certains moments de la déformation, enfin nous tracerons les erreurs obtenues en fonction du temps.

TEST 1

La figure FIG.2.3 représente la courbe déformée, formée de 2 courbes PH raccordées G1.

Nous positionnons 12 capteurs régulièrement répartis, et nous appliquons les 2 algorithmes en même temps. Quelques résultats visuels sont représentés dans le tableau de figures TAB.2.1.

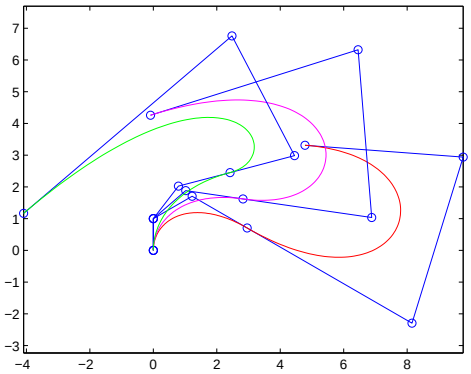
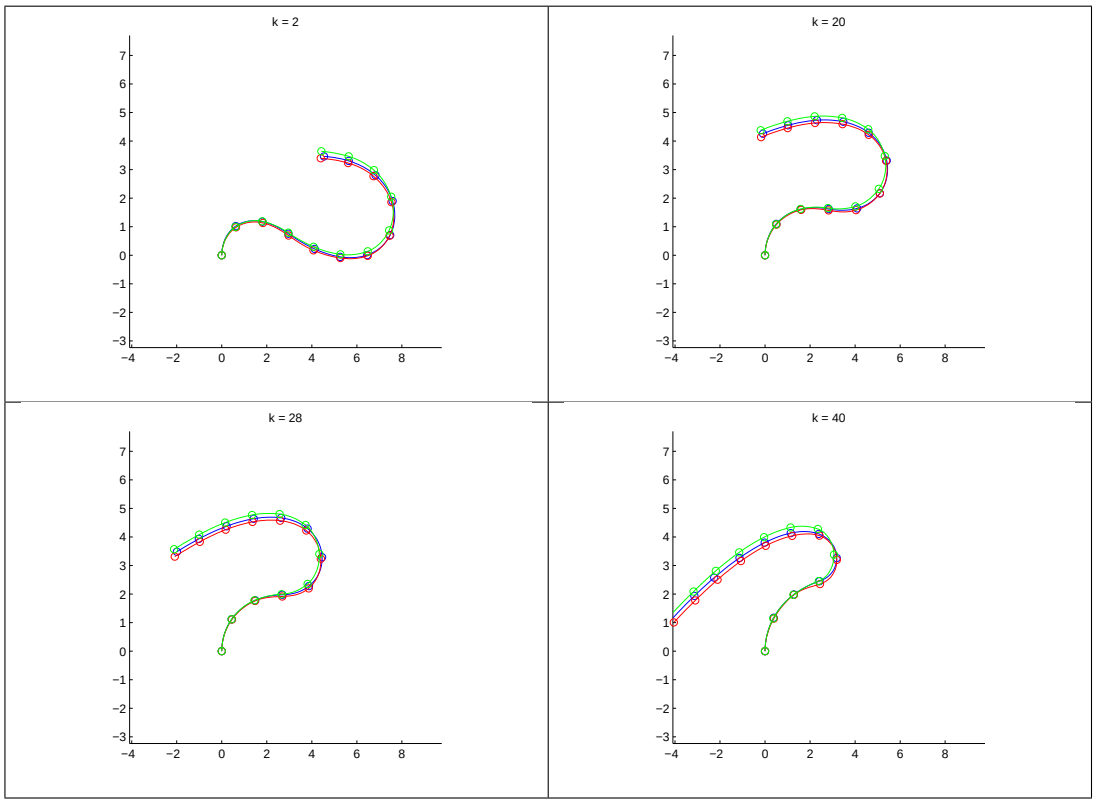


FIG. 2.3 – Test 1, *(rouge) : courbe de départ, (rose) : courbe intermédiaire, (vert) : courbe d'arrivée*



TAB. 2.1 – Etapes de reconstruction pour le test 1, *(bleu) : courbes initiales, (rouge) : courbes reconstruites par la méthode d'interpolation, (vert) : courbes reconstruites par la méthode de déformation*

Les erreurs calculées sont les suivantes : (voir FIG.2.4)

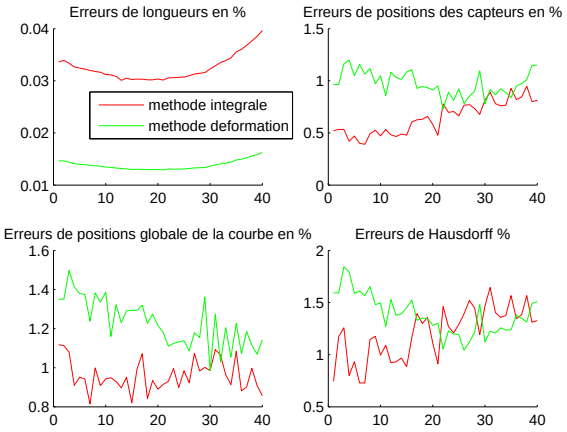


FIG. 2.4 – Test 1 : erreurs

TEST 2

La figure FIG.2.5 représente la courbe déformée.

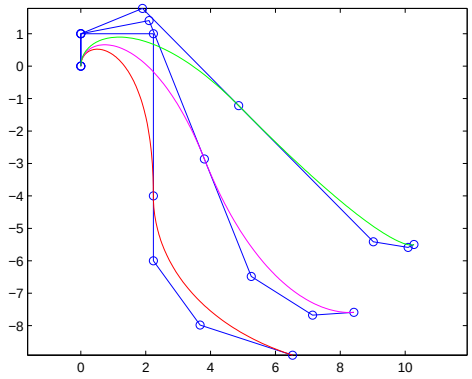
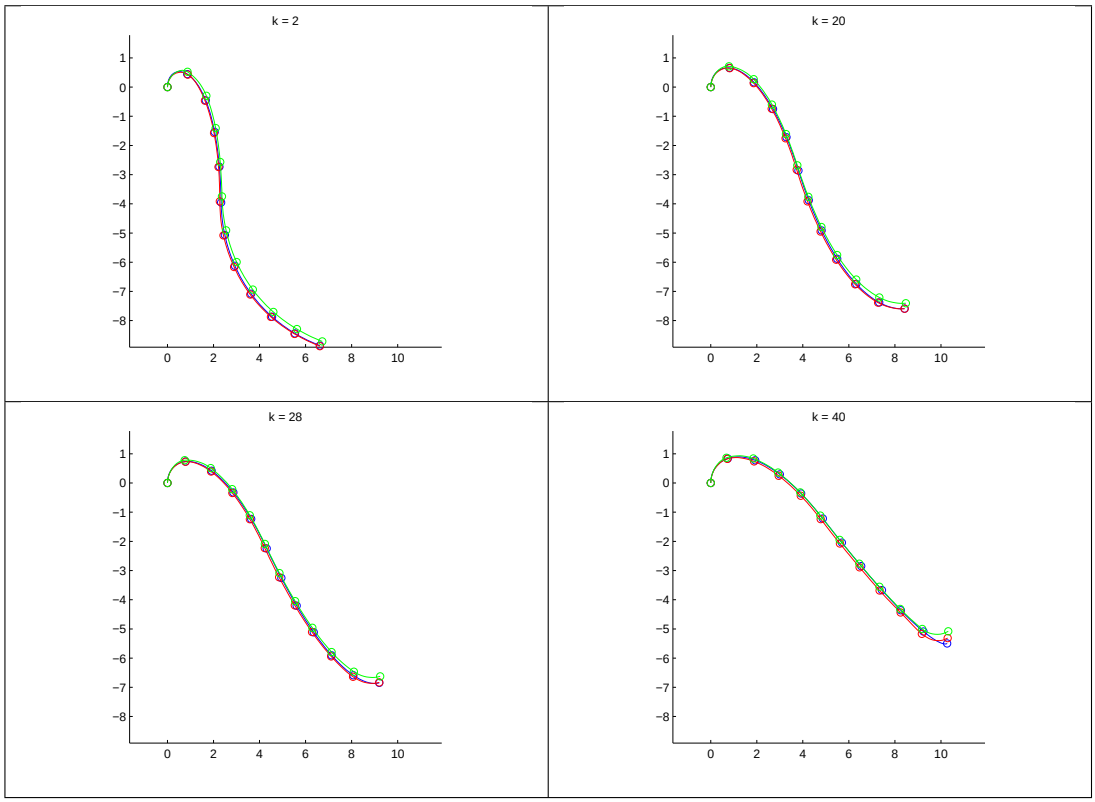


FIG. 2.5 – Test 2 : courbes, *(rouge) : courbe de départ, (rose) : courbe intermédiaire, (vert) : courbe d'arrivée*

Nous positionnons 12 capteurs régulièrement répartis, et nous appliquons les 2 algorithmes en même temps. Les résultats sont dans le tableau de figures TAB.2.2.

Les erreurs calculées sont représentées dans la figure FIG.2.6.



TAB. 2.2 – Etapes de reconstruction pour le test 2, *(bleu) : courbes initiales, (rouge) : courbes reconstruites par la méthode d'interpolation, (vert) : courbes reconstruites par la méthode de déformation*

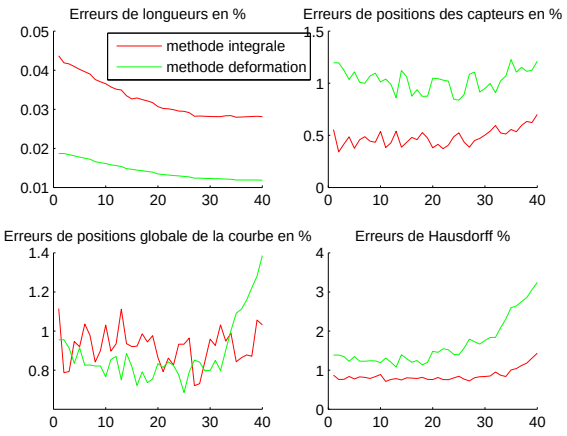


FIG. 2.6 – Test 2 : erreurs

TEST 3

La figure FIG.2.7 représente la courbe déformée.
Nous positionnons 12 capteurs régulièrement répartis, et nous appliquons les 2 algo-

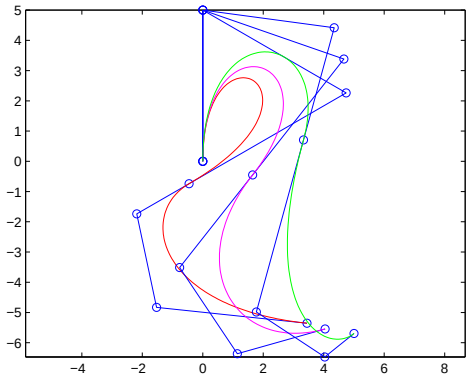
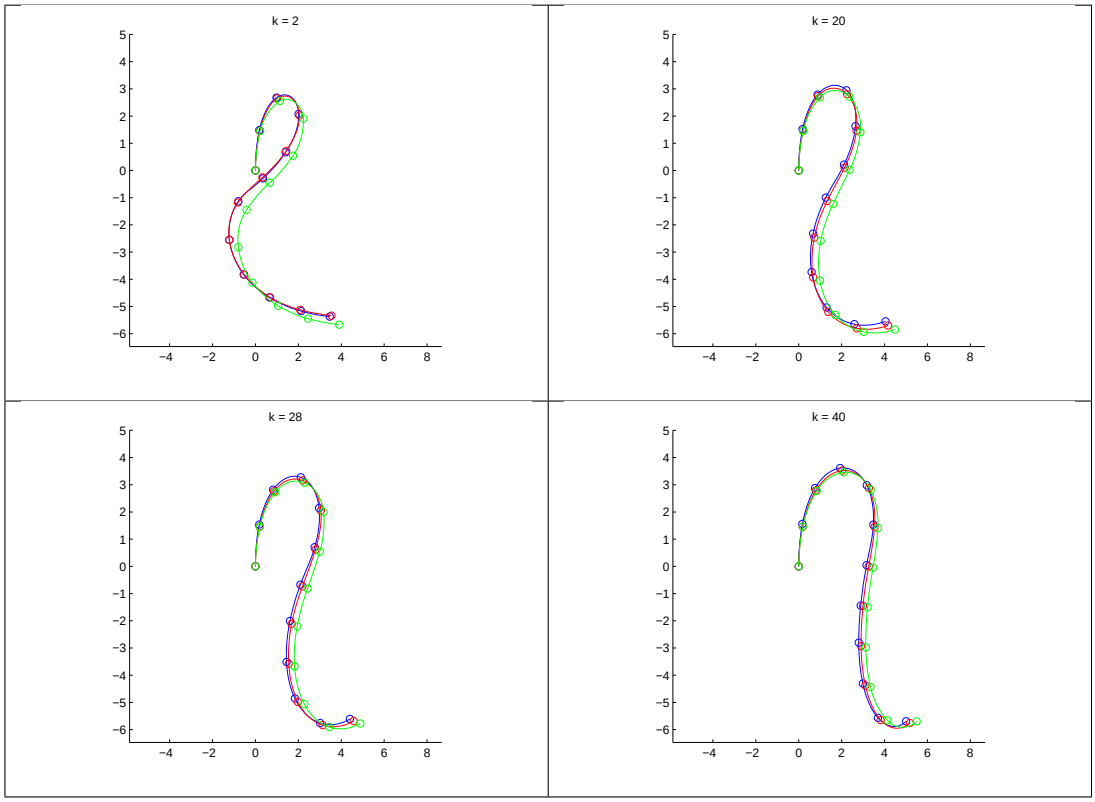


FIG. 2.7 – Test 3 : courbes, *(rouge) : courbe de départ, (rose) : courbe intermédiaire, (vert) : courbe d'arrivée*

rithmes en même temps. Voyons quelques résultats visuels (voir TAB.2.3).



TAB. 2.3 – Etapes de reconstruction pour le test 3, *(bleu) : courbes initiales, (rouge) : courbes reconstruites par la méthode d'interpolation, (vert) : courbes reconstruites par la méthode de déformation*

Les erreurs calculées sont reportées dans la figure FIG.2.8.

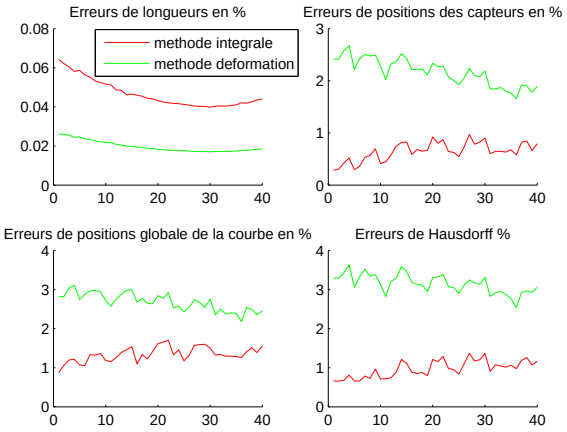


FIG. 2.8 – Test 3 : erreurs

COMMENTAIRES

Ces trois exemples montrent que nous obtenons de bonnes reconstructions avec les deux méthodes, avec des erreurs acceptables dans les deux cas, mais en général, des erreurs légèrement plus faibles pour la première méthode développée (sauf en ce qui concerne l'erreur de longueur). Le prochain exemple montre un avantage à la deuxième méthode.

TEST 4

Voyons ici un exemple qui illustre un inconvénient de la première méthode développée : il illustre une forte courbure et un nombre faible de capteurs qui ne peut alors fournir les informations concernant cette forme de la courbe (nous pouvons cependant voir dans cet exemple l'aspect assez caricatural de la déformation).

La figure FIG.2.9 représente la courbe déformée.

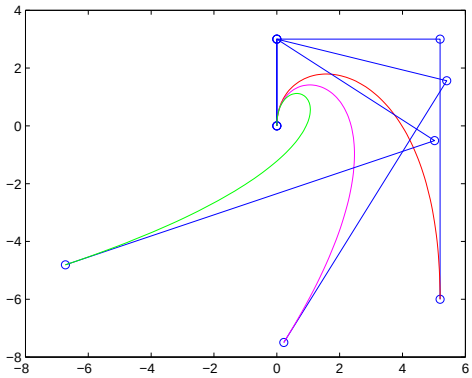
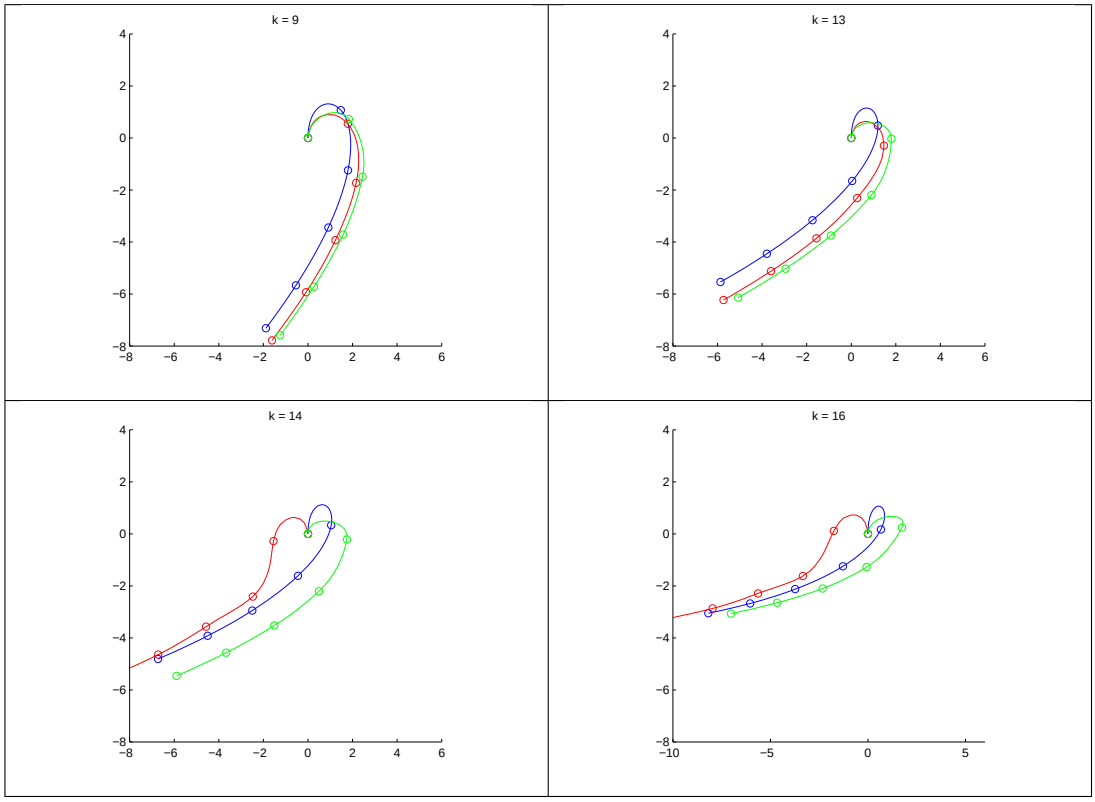


FIG. 2.9 – Test 4 : courbes, *(rouge) : courbe de départ, (rose) : courbe intermédiaire, (vert) : courbe d'arrivée*

Nous positionnons 6 capteurs régulièrement répartis, et nous appliquons les 2 algo-

rithmes en même temps. Voyons quelques résultats visuels dans le tableau TAB.2.4.



TAB. 2.4 – Etapes de reconstruction pour le test 4, *(bleu) : courbes initiales, (rouge) : courbes reconstruites par la méthode d'interpolation, (vert) : courbes reconstruites par la méthode de déformation*

Voici les erreurs calculées (voir FIG.2.10)

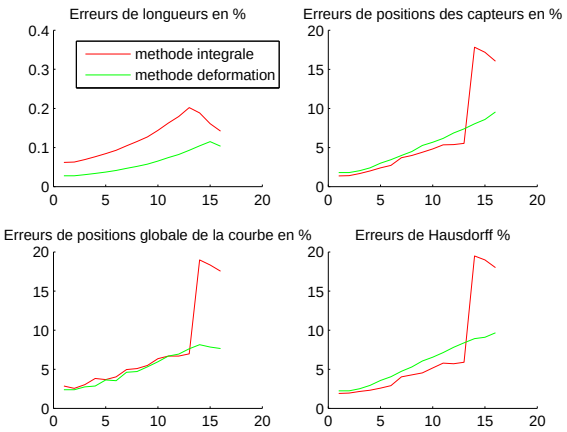


FIG. 2.10 – Test 4 : erreurs

Nous voyons apparaître un problème de reconstruction avec la première méthode à

partir de l'étape 14. Cela montre en fait que les deux méthodes sont pénalisées par la forte courbure que prend la courbe entre deux points de mesure. Cela est dû au fait que pour la première méthode, nous interpolons les angles, qui sont connus à 2π près, et nous avons supposé que deux angles avaient une différence maximale de π, mais aussi que la méthode donne un sens physique aux courbes qu'elle crée, sens physique qui n'est pas ressenti pour cette courbe à cause du nombre insuffisant de capteurs pour en rendre compte. La seconde méthode qui utilise la forme au temps précédent ne rencontre pas ce genre de problèmes, même s'il n'est pas possible d'obtenir des courbes aussi courbées avec des fonctions de Hermite (et un nombre aussi faible de points de mesure).

COMMENTAIRES

En conclusion, nous pouvons dire que la première méthode développée, qui interpole la fonction des angles et intègre, sans tenir compte de la position au temps précédent est meilleure. Cependant, lorsqu'il n'y pas beaucoup de capteurs, et que la courbe ne satisfait pas l'hypothèse que nous avons supposée pour la différence entre les angles successifs, la deuxième méthode, qui est une méthode de suivi de forme, garde la forme globale de la courbe. Nous pourrions cependant pour la première méthode modifier la valeur des angles en fonction de leur valeur au temps précédent, cela permettrait alors d'améliorer ces résultats.

Voyons à présent quelques résultats sur les courbes gauches, qui ne feront pas apparaître ce genre de problèmes car nous travaillons directement sur les vecteurs tangents dans l'espace pour les deux méthodes.

2.2.2 TESTS COURBES GAUCHES

Nous allons ici tester les algorithmes sur des courbes gauches en déformation simulées. Nous avons également pris des courbes tests sous forme de courbes PH cubiques gauches, car il est assez aisé de les déformer en gardant la longueur constante (les détails sont en annexe C).

TEST 1

Voici la séquence de courbe en déformation (FIG.2.11). Nous voyons la courbe de départ en rouge avec son polygone de contrôle et la courbe d'arrivée en vert avec son polygone de contrôle, ainsi qu'une courbe intermédiaire en rose. Nous avons également dessiné les projections sur trois plans orthogonaux afin de mieux visualiser les courbes dans l'espace.

Nous positionnons 15 capteurs régulièrement répartis, et nous appliquons les 2 algorithmes en même temps. Quelques résultats visuels sont rassemblés dans le tableau TAB.2.5, où nous voyons superposés les courbes simulées en bleu, les courbes reconstruites par la méthode intégrale en rouge et la méthode de reconstruction par déformation en vert. Les ronds sont les positions des capteurs.

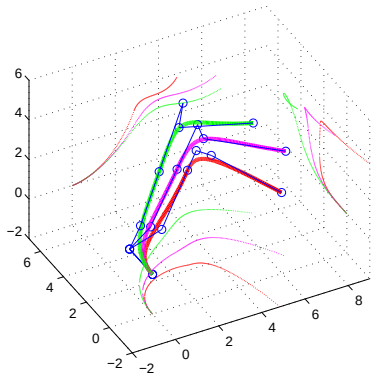
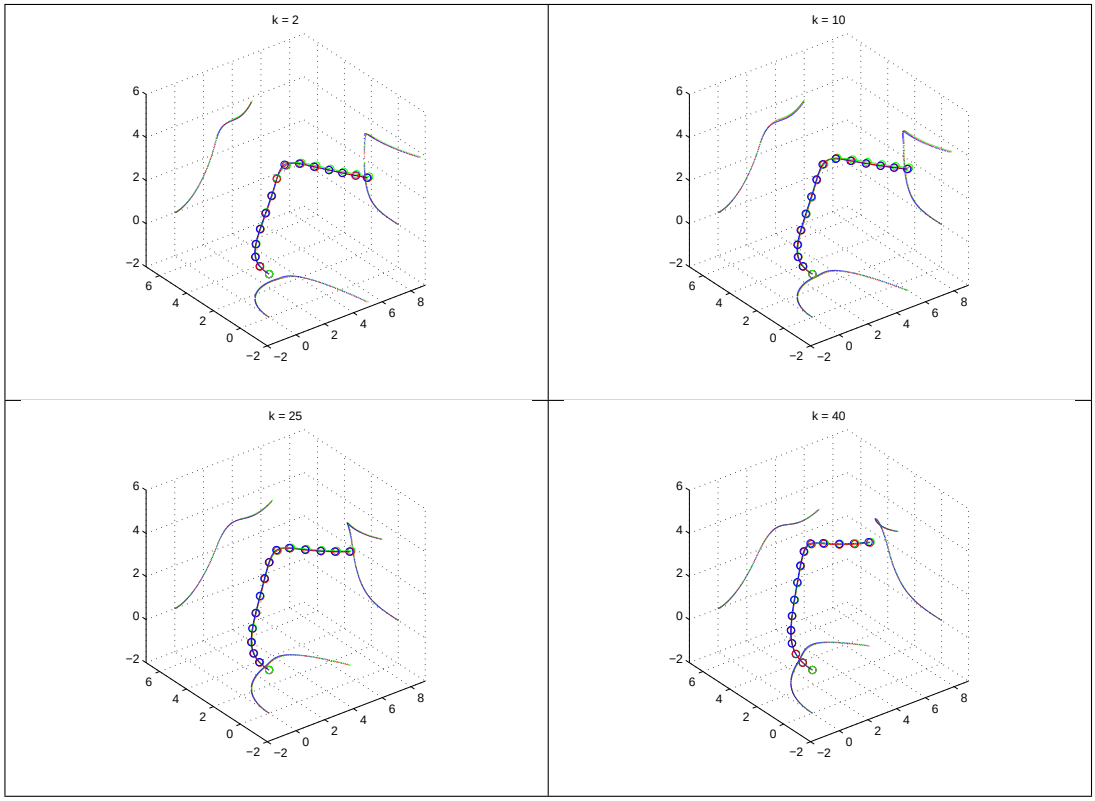


FIG. 2.11 – Test 1 : courbes, *(rouge) : courbe de départ, (rose) : courbe intermédiaire, (vert) : courbe d'arrivée*



TAB. 2.5 – Etapes de reconstruction pour le test 1, *(bleu) : courbes initiales, (rouge) : courbes reconstruites par la méthode d'interpolation, (vert) : courbes reconstruites par la méthode de déformation*

Les deux reconstructions sont très proches visuellement des courbes de référence. Voyons les erreurs obtenues au cours du temps par ces méthodes sur la figure FIG.2.12.

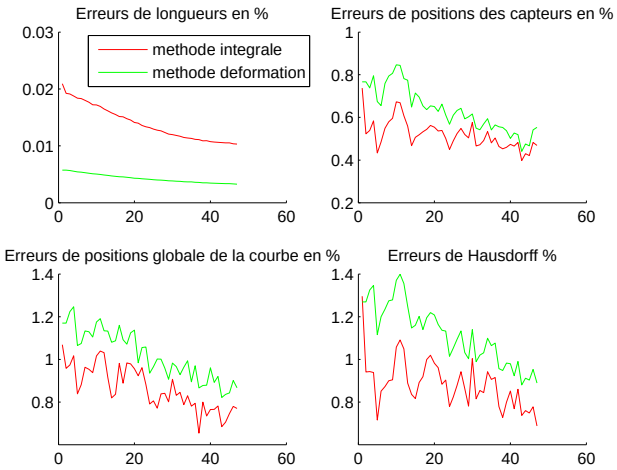


FIG. 2.12 – Test 1 : erreurs

Ces graphes d'erreur montrent un même comportement des méthodes, avec des erreurs faibles et du même ordre au cours du temps.

TEST 2

Voici la séquence de courbe en déformation (voir FIG.2.13).

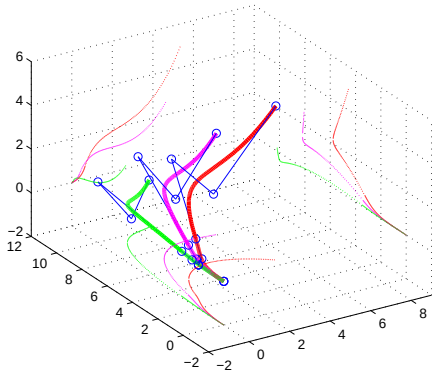
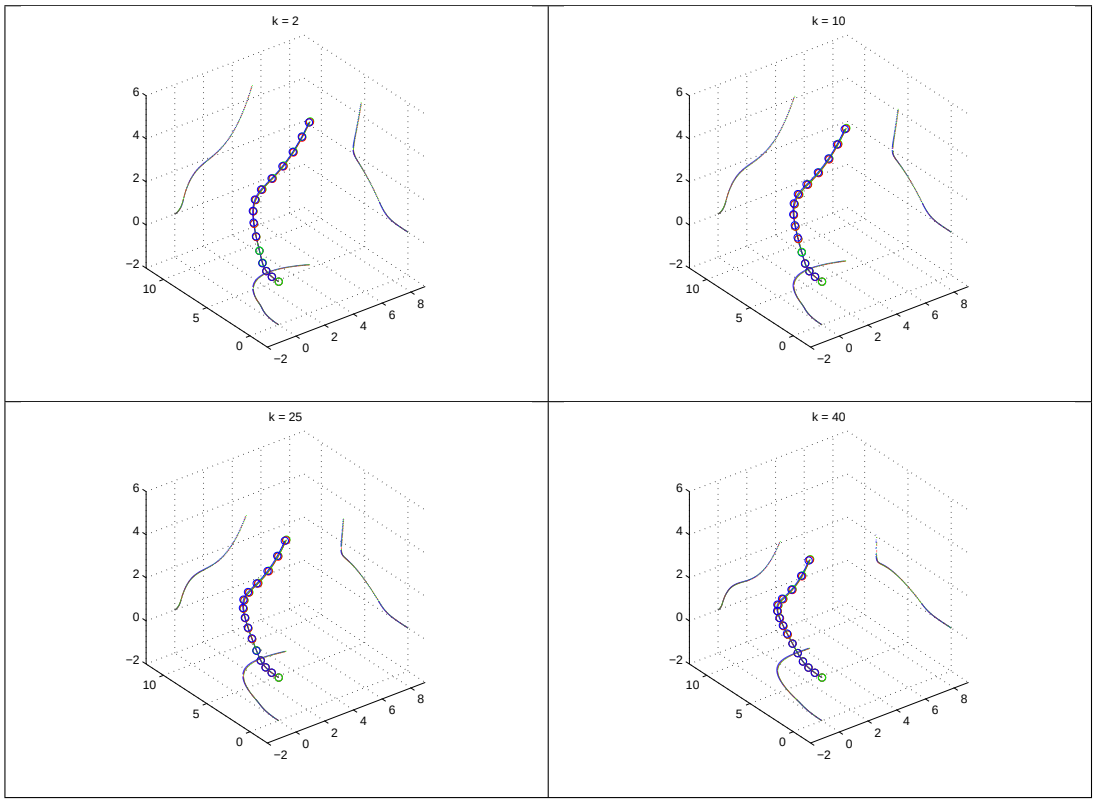


FIG. 2.13 – Test 2 : courbes, *(rouge) : courbe de départ, (rose) : courbe intermédiaire, (vert) : courbe d'arrivée*

Nous positionnons 15 capteurs régulièrement répartis, et nous appliquons les 2 algorithmes en même temps. Voyons quelques résultats visuels rassemblés dans le tableau TAB.2.6. La courbe simulée est en bleu, la courbe obtenue avec la méthode d'intégration en rouge et celle obtenue par déformation est en vert.



TAB. 2.6 – Etapes de reconstruction pour le test 2, *(bleu) : courbes initiales, (rouge) : courbes reconstruites par la méthode d'interpolation, (vert) : courbes reconstruites par la méthode de déformation*

Nous obtenons également pour cet exemple des courbes reconstruites visuellement très proches des courbes initiales. Les erreurs calculées sont visualisées sur la figure FIG.2.14.

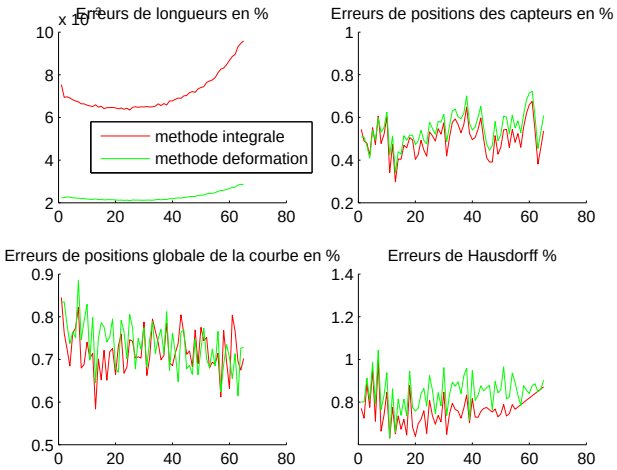


FIG. 2.14 – Test 2 : erreurs

Nous obtenons des erreurs similaires et très faibles pour les deux méthodes : les deux méthodes sont équivalentes ici.

2.3 CONCLUSION

La perspective de reconstruire des déformations de courbes dans le temps nous a permis de développer une nouvelle méthode qui utilise la position de la courbe au temps précédent, ce que ne faisait pas du tout la méthode précédente, et qui pouvait présenter un handicap lors de cette évolution de reconstruction dans le temps. La nouvelle méthode élaborée consiste à trouver pas à pas les déplacements des positions des capteurs avec les contraintes de tangentes (qui sont écrites dans le modèle car nous utilisons la modélisation avec des courbes de Hermite), et des contraintes de longueur. Cette méthode consiste en une succession de minimisations sous contraintes.

Les résultats comparatifs montrent que les deux méthodes donnent de très bons résultats. Il nous serait possible d'améliorer les deux méthodes en modifiant certains paramètres ou en augmentant l'échantillonnage de nos courbes reconstruites, que nous avons limité pour obtenir une rapidité des algorithmes.

De même, nous pourrions choisir entre les deux méthodes en fonction du temps de calcul. En l'état actuel, la méthode de déformation a un temps de calcul du même ordre que la méthode d'interpolation, mais nous n'avons en rien essayé d'optimiser les calculs, donc de nombreuses améliorations sont possibles de ce point de vue également.

Enfin, cette méthode apporte des outils de calcul pour travailler dans l'espace direct (contrairement à la méthode d'interpolation de la fonction dérivée qui travaillait dans l'espace tangentiel), et cela va nous être utile dans la résolution de l'étape suivante : la reconstruction de surfaces.

Deuxième partie

Reconstruction de surfaces

Après la reconstruction des courbes, nous nous intéressons désormais à la reconstruction des surfaces. En premier lieu, notons qu'il n'existe pas pour les surfaces de paramétrisation *canonique* équivalente à la paramétrisation en abscisse curviligne des courbes. Hormis peut-être pour les surfaces développables.

Ainsi, notre réflexion a tout d'abord porté sur la définition d'un choix de modèle surfacique pour la reconstruction en lien avec le contexte technologique d'acquisition lié à nos capteurs, avec en particulier la question de la structuration topologique de nos capteurs sur la surface afin d'obtenir des données cohérentes pour la reconstruction. Ce qui nous a conduit à plusieurs stratégies d'acquisition/reconstruction.

Une topologie triangulaire des capteurs supposait une méthodologie d'acquisition complexe tant du point de vue technologique que mise en œuvre pratique. De sorte que, fondamentalement, il nous est apparu très vite la nécessité de s'appuyer sur la structure linéaire de notre ruban, c'est-à-dire sur la notion de courbe tracée sur la surface qui repose sur une topologie *tensorielle* (c'est-à-dire en rectangle) du processus tant d'acquisition que de reconstruction. Cette approche prolonge naturellement le travail des chapitres précédents et, d'un point de vue expérimental, ne nécessite pas de prototype nouveau, car elle s'appuie sur le même ruban de capteurs.

Partant de là, plusieurs stratégies et modèles pour l'acquisition et la reconstruction surfacique sont envisageables.

• Une première idée de système de capture de surfaces consiste en un *filet* instrumenté, les capteurs étant aux différentes intersections des mailles. La connaissance de la surface passe donc par la connaissance du double réseau de courbes que forme le filet. Idéalement, ces courbes devraient être représentatives de caractéristiques géométriques de la surface à reconstruire, telles que des lignes de courbure ou des géodésiques de cette surface. Ces caractéristiques seraient alors prises en compte dans le modèle surfacique de reconstruction. On comprendra néanmoins aisément que la connaissance/acquisition de telles courbes spécifiques d'une surface que nous cherchons justement à déterminer est une gageure.

Remarquons cependant qu'un tel filet de données, supposé bien évidemment " tendu " peut s'assimiler à une surface développable. De sorte que les courbes composant ce filet définissent des géodésiques de cette surface/filet. Ainsi, si notre surface à reconstruire est connue comme étant développable, la mise en place du filet sur cette surface induira un quadrillage de la surface par un réseau de courbes géodésiques (avec des inter-distances connues entre les nœuds du maillage).

Ainsi, dans le cas général, nous avons développé un modèle de reconstruction surfacique s'appuyant sur cette stratégie de " filet de données " à partir de données numériques issues de courbes isoparamétriques de surfaces tests quelconques et à partir de données expérimentales issues de surfaces développables. Ce modèle de reconstruction s'appuyant sur les trois étapes suivantes : reconstructions des courbes isoparamétriques, fermeture de ce réseau, puis remplissage pour reconstituer la surface.

• Une deuxième stratégie pour la capture de surface s'appuie sur une seule famille de courbes dans une seule direction à l'aide du ruban de capteurs. Cette approche évite la phase de fermeture du réseau. Ce modèle surfacique de reconstruction a été testé à l'aide de courbes isoparamétriques et de courbes géodésiques sur des surfaces numériques tests, partant d'un même bord de la surface.

Cette approche plus simple se justifie pleinement si l'on remarque que l'instrumentation de surfaces réelles à l'aide de notre ruban de capteurs fournit réellement des géodésiques sur

cette surface. En effet, notre ruban instrumenté étant de faible épaisseur, son application continue et tangentielle sur la surface le conduit à suivre des courbes géodésiques sur cette surface (voir[HEN98], page 69, The Ribbon Test). Ainsi, la connaissance de cette caractéristique des courbes issues du ruban de capteur nous permettra finalement de proposer une méthode spécifique de reconstruction.

La structure de cette deuxième partie est identique à celle suivie pour la reconstruction des courbes. Dans un premier chapitre, nous présentons la stratégie de reconstruction des surfaces en statique. Une méthodologie globale sera présentée, puis nous proposerons différents modèles, avec leur mise en œuvre, selon les données dont nous disposons : filet de données ou courbes dans une seule direction. Ces méthodes seront ensuite comparées au travers d'exemples. Dans un deuxième chapitre, nous nous intéressons au suivi de trajectoire, toujours avec les deux types de données possibles.

Chapitre 3

Reconstruction de surfaces

Dans ce chapitre, nous traitons la question de la reconstruction de surfaces à partir de données tangentielles. Nous nous sommes en premier lieu posé la question de l'organisation structurelle des capteurs, et nous avons fait le choix de garder une structure filaire en conservant les modèles de prototypes étudiés dans la partie précédente. A partir de là, deux modèles de systèmes d'acquisition sont possibles : organiser les rubans de capteurs dans deux directions complémentaires, et obtenir ainsi un filet de capteurs, ou bien utiliser les rubans dans une seule direction. La méthode de reconstruction de surfaces à partir de ces données se structure alors en deux étapes, la première étant de reconstruire un réseau de courbes satisfaisant les données acquises, la deuxième étant de définir une surface s'appuyant sur ces courbes. Nous verrons dans un premier temps comment il nous est possible de créer un réseau de courbes à partir des données (et dépendant du système d'acquisition), puis comment nous définissons la surface solution.
Nous comparons dans un second temps sur différents exemples les diverses méthodes développées, en nous focalisant d'abord sur les réseaux de courbes reconstruits, puis sur les surfaces reconstruites. Enfin, nous étudions le comportement de certaines erreurs de reconstruction en fonction du nombre de capteurs utilisés, afin de juger du caractère convergent des méthodes employées.

Sommaire

3.1 DONNÉES

Nous allons nous intéresser ici à la reconstruction de surfaces à partir d'un réseau de capteurs. Comme nous venons de le voir dans l'introduction précédente, nous avons deux possibilités de systèmes de capture de formes : avoir un filet de capteurs d'une part, ou bien avoir une famille de rubans intrumentés dans la même direction d'autre part. Nous allons ici expliquer les deux systèmes de données que nous obtenons, puis nous homogénéiserons afin d'obtenir une modélisation globale avec des contraintes différentes.

• Considérons tout d'abord le premier modèle de capture de mouvement : **le filet de données**. Lorsque nous cherchons à reconstruire une surface, nous utilisons donc un système de mesure ayant la forme d'un filet, c'est-à-dire un réseau fermé de courbes s'intersectant, avec des points de mesure en ces intersections. Nous pouvons représenter les données grâce à la figure FIG.3.1.

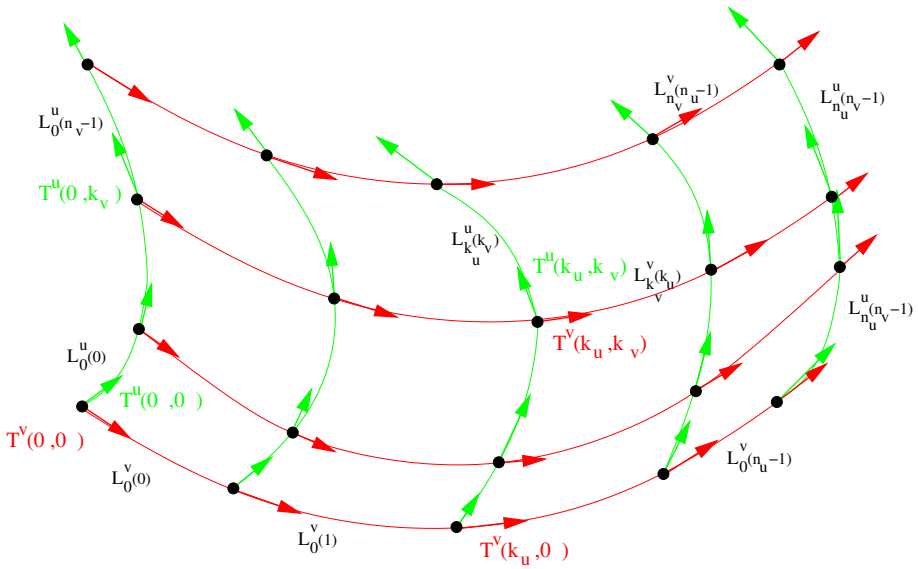


FIG. 3.1 – Données acquises par le filet

Nous supposons donc que pour un filet constitué de $(n_u+1).(n_v+1)$ capteurs, nous avons les données suivantes :

– les données tangentielles des courbes dans les deux directions aux intersections du réseau $(k_u,k_v), k_u=0,...n_u, k_v=0,...,n_v$:

$$T^u(k_u,k_v), k_u=0,...,n_u, k_v=0,...,n_v$$

dans une direction et

$$T^v(k_u,k_v), k_u=0,...,n_u, k_v=0,...,n_v$$

dans la direction complémentaire,

– les distances curvilignes le long de ces courbes entre capteurs adjacents :

$$L^u_{k_u}(k_v), k_u = 0, ..., n_u; k_v = 0, ..., n_v - 1$$

étant la distance curviligne entre les capteurs (k_u, k_v) et $(k_u, k_v + 1)$ le long de la courbe numérotée k_u dans la première direction, et

$$L^v_{k_v}(k_u), k_v = 0, ..., n_v; k_u = 0, ..., n_u - 1$$

étant la distance curviligne entre les capteurs (k_u, k_v) et $(k_u + 1, k_v)$ le long de la courbe k_v dans la direction complémentaire.

Nous devons alors reconstruire une surface qui contient deux familles de courbes s'intersectant dont nous connaissons les contraintes tangentielles et la répartition des points de mesure dans les deux directions.

• Voyons à présent les données acquises par le deuxième système de mesure : **une famille de "rubans intrumentés" dans la même direction**. Voyons schématiquement sa représentation sur la figure FIG.3.2. Les rubans instrumentés sont représentés en rouge, et en des points de répartition connue le long de ces courbes, nous avons des connaissances tangentielles dans deux directions : le long des courbes mais également la direction orthogonale à ces courbes sur la surface.

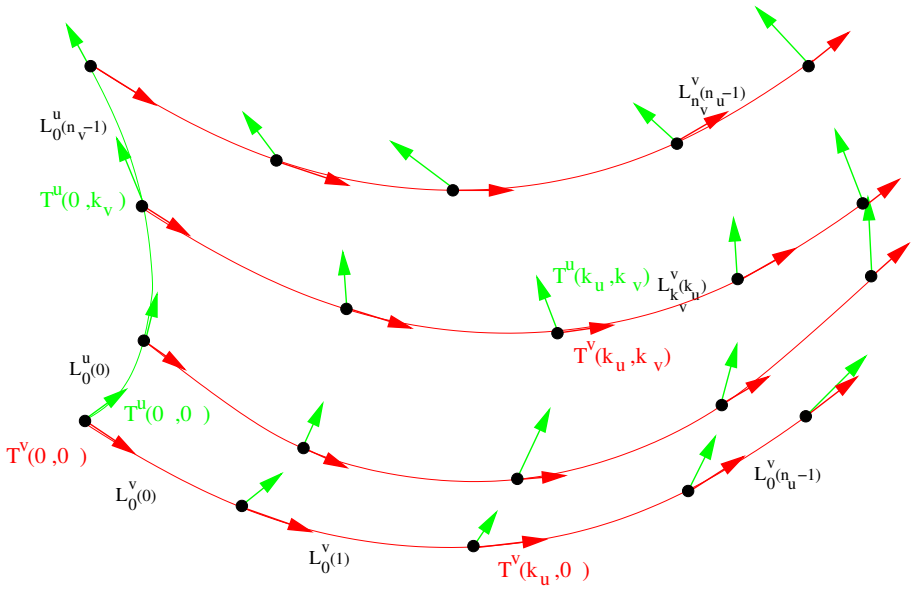


FIG. 3.2 – Données acquises par une seule famille de courbes

Dans ce cas, nous connaissons la répartition des capteurs dans une seule direction (excepté pour les points de départ de toutes ces courbes qui doivent être reliées avec également une connaissance de la répartition) et nous avons des données tangentielles non seulement dans la direction des courbes instrumentées, mais également dans une direction complémentaire. Nous voyons alors que nous pouvons créer un ensemble de courbes virtuelles dans

l'autre direction, dont nous connaitrions les données tangentielles en certains points mais dont nous ne connaitrions pas les contraintes de longueur entre ces capteurs. Voyons ce réseau fermé virtuellement dans la figure FIG.3.3 dans lequel des courbes vertes en pointillé ont été ajoutées.

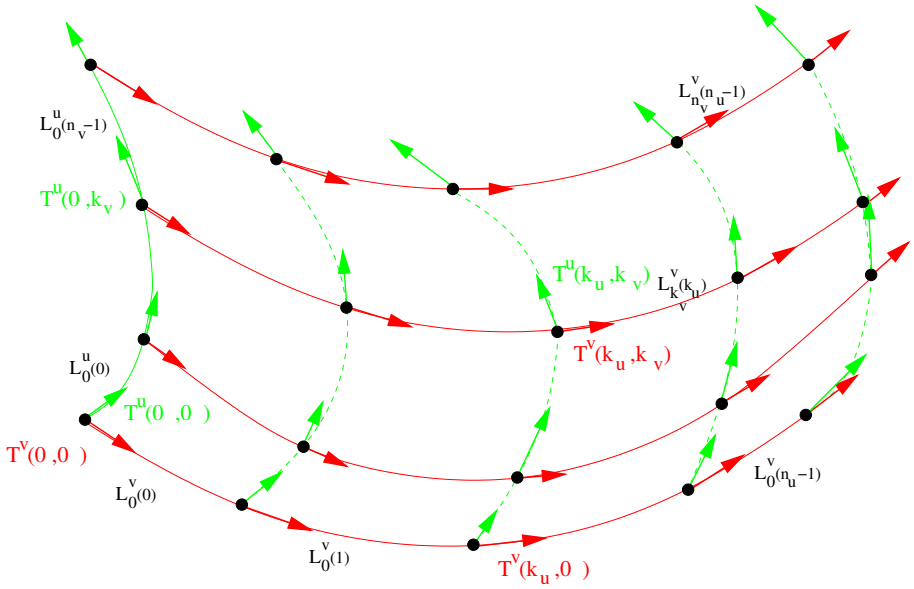


FIG. 3.3 – Réseau fermé virtuel associé

Nous avons ainsi également un réseau fermé de courbes à reconstruire dont les données sont :

- les données des capteurs dans les deux directions de courbes :

$$T^u(k_u,k_v), k_u = 0,...,n_u, k_v = 0,...,n_v$$

et

$$T^v(k_u,k_v), k_u = 0,...,n_u, k_v = 0,...,n_v,$$

- les distances curvilignes le long des courbes dans une seule direction entre capteurs adjacents :

$$L^v_{k_v}(k_u), k_v = 0,...,n_v; k_u = 0,...,n_u - 1$$

étant la distance curviligne entre les capteurs (k_u,k_v) et (k_u+1,k_v) le long de la courbe k_v de la deuxième direction, et les distances curvilignes pour la courbe qui relie les points de départ de la première famille de courbes :

$$L^u_0(k_v), k_v = 0,...,n_v - 1$$

Ainsi, nous voyons que dans les deux cas, notre surface est déterminée par des données sur un réseau de courbes, c'est-à-dire deux familles de courbes s'intersectant. Nous avons soit des données tangentielles et des contraintes de longueur dans les deux directions (en ce qui concerne les données fournies par un filet de capteurs), soit des contraintes tangentielles dans les deux directions, mais des contraintes de longueur dans une seule (pour ce qui est des rubans instrumentés dans un seul sens).

3.2 MÉTHODOLOGIE GLOBALE

Les surfaces étant connues à partir d'un réseau de courbes, voici la méthodologie que nous allons employer :

- dans un premier temps nous devons reconstruire le réseau de courbes formant la structure de la surface (cette étape diffère selon les données que nous avons),
- puis nous devons créer une surface s'appuyant sur cette structure.

La surface sera reconstruite sous forme paramétrique :

$$S(u,v) = \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix}, (u,v) \in [0,1] \times [0,1], \tag{3.1}$$

ce qui signifie que les données des capteurs seront supposées connues aux intersections des isoparamétriques :

$$P_{k_u,k_v} = S\left(\frac{k_u}{n_u}, \frac{k_v}{n_v}\right), \ k_u = 0, ..., n_u, k_v = 0, ..., n_v. \tag{3.2}$$

Voyons maintenant comment nous arrivons à déterminer une paramétrisation de cette surface.

La première étape consiste à reconstruire deux réseaux de courbes gauches interprétés comme les isoparamétriques de la surface.

$$C^u_{k_u}(v) = S\left(\frac{k_u}{n_u}, v\right), k_u = 0, ..., n_u \tag{3.3}$$

sont les courbes isoparamétriques en u et

$$C^v_{k_v}(u) = S\left(u, \frac{k_v}{n_v}\right), k_v = 0, ..., n_v \tag{3.4}$$

sont les courbes isoparamétriques en v.

Une illustration de cette paramétrisation est visible sur la figure FIG.3.4.

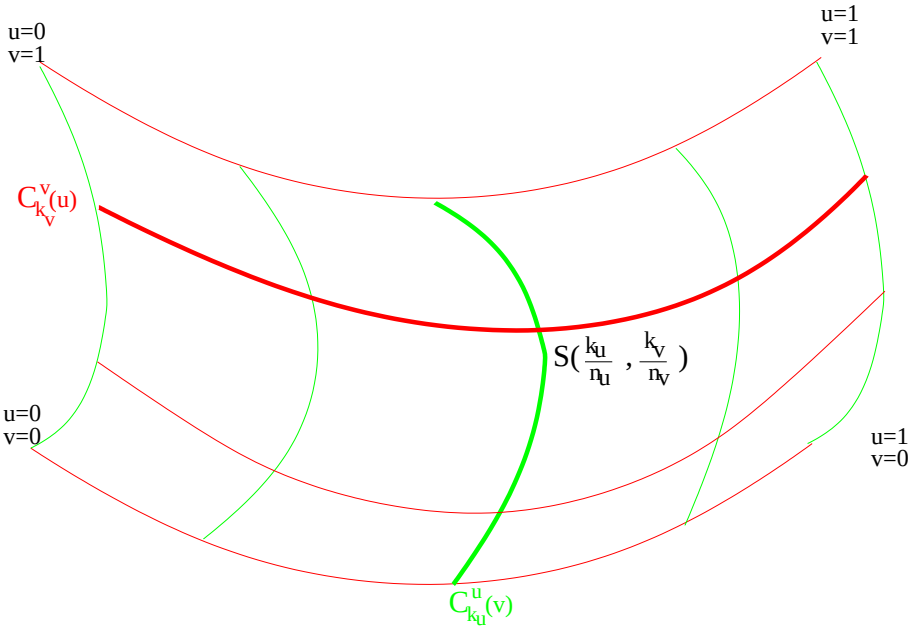


FIG. 3.4 – Réseau des courbes isoparamétriques

Nous ne pouvons pas employer une méthode unique pour la reconstruction de ce réseau de courbes, car selon le type de système de capture, nous n'avons pas accès aux mêmes données. Nous allons donc traiter le problème séparément selon le cas de figure dans lequel nous nous trouvons.

3.3 PREMIER CAS : LES DONNÉES SONT UN FILET

3.3.1 RÉSEAU INITIAL

Dans le cas de figure d'un filet de données, nous avons des contraintes tangentielles et des contraintes de longueur pour les deux familles de courbes, et ces courbes doivent s'intersecter en les points de mesure.

Les courbes $C^u_{k_u}(v)$ isoparamétriques en u sont déterminées à partir des capteurs (k_u, j) pour $j = 0, ..., n_v$ et les distances inter-capteurs $L^u_{k_u}(j), j = 0, ..., n_v - 1$. Les courbes $C^v_{k_v}(u)$ isoparamétriques en v sont quant à elles déterminées à partir des capteurs (i, k_v) pour $i = 0, ..., n_u$ et les distances inter-capteurs $L^v_{k_v}(i), i = 0, ..., n_u - 1$. Ces données nous permettent d'utiliser la méthode mise en place pour la reconstruction des courbes gauches à partir de données tangentielles vue dans la première partie.

Nous pouvons noter que nous devons obtenir des courbes dont le paramètre a ses valeurs dans $[0, 1]$. Dans ce cas, nous utiliserons d'abord notre algorithme (nous obtenons

alors des courbes dont le paramètre est l'abscisse curviligne, c'est-à-dire variant de 0 à la longueur de la courbe), et nous devrons ensuite faire un changement de variable sur chaque courbe pour obtenir notre paramètre entre 0 et 1.

Nous devons connaître le point de départ de chaque courbe pour la construire par rapport aux autres, nous devons donc les calculer dans un ordre précis : nous commençons par construire $C_0^u(v)$, nous connaissons ainsi tous les points de départ des isoparamétriques en v, les courbes $C_{k_v}^v(u)$:

$$C_{k_v}^v(0) = C_0^u\left(\frac{k_v}{n_v}\right) \tag{3.5a}$$

et grâce à $C_0^v(u)$, nous connaissons enfin les points de départ des isoparamétriques en u, les courbes $C_{k_u}^u(v)$:

$$C_{k_u}^u(0) = C_0^v\left(\frac{k_u}{n_u}\right) \tag{3.5b}$$

Nous avons alors toutes nos courbes reconstruites les unes par rapport aux autres.

Nous obtenons ainsi une grille de courbes gauches, caractéristiques de la surface modulo notre filet de capteurs et nous devons trouver une surface qui s'appuie sur ces courbes. La surface pourra être déterminée si et seulement si le réseau de courbes forme un vrai réseau, c'est-à-dire où les courbes s'intersectent (sinon nous ne trouverons pas de surface continue qui pourra s'appuyer sur ces courbes). Or nos courbes sont construites de façon indépendante (aucune contrainte de passage en des points d'intersection) donc nous devons modifier ce réseau.

3.3.2 RECALAGE DU RÉSEAU

Nous devons modifier les courbes existantes de sorte qu'elles s'intersectent aux points capteurs, c'est-à-dire que nous devons avoir :

$$C_{k_u}^u\left(\frac{k_v}{n_v}\right) = C_{k_v}^v\left(\frac{k_u}{n_u}\right), k_u = 1, ..., n_u, k_v = 1, ..., n_v \tag{3.6}$$

les autres points de croisement étant des points d'intersection par construction (les points qui se situent sur des courbes de bord pour $k_u = 0$ ou $k_v = 0$).

De plus, nous devons conserver les tangentes en ces points-là (données des capteurs) ainsi que les contraintes de longueur entre les points capteurs.

Deux stratégies ont été développées :

- modification des courbes obtenues précédemment en forçant les points de passage, cela modifiera légèrement les longueurs,
- définition de nouvelles courbes, proches des anciennes en cherchant les points de passage tels que les longueurs soient conservées.

A. MODIFICATION DES COURBES EXISTANTES

Le premier réseau obtenu donne un ensemble de courbes proches des courbes voulues, le seul problème étant qu'il faille modifier légèrement les courbes afin qu'elles s'intersectent. Nous pouvons ainsi penser que si nous imposons des points d'intersection, ils seront proches des courbes initiales et donc leur modification n'engendrera pas de grands écarts de longueur (nous devons conserver la longueur des courbes car c'est une donnée du système). Voici donc la premier méthode de modification du réseau : imposer les points d'intersection, et modifier les courbes pour qu'elles passent par ces points.

A.a. Première méthode Pour cela, nous voyons deux façons différentes de procéder. La première possibilité consiste à déterminer dès le début tous les points d'intersection du réseau (prenons par exemple tous les points milieux) :

$$P_{k_u,k_v} = \frac{1}{2}\left(C^u_{k_u}\left(\frac{k_v}{n_v}\right) + C^v_{k_v}\left(\frac{k_u}{n_u}\right)\right), k_u = 1, ..., n_u, k_v = 1, ..., n_v \tag{3.7}$$

puis nous devons modifier les courbes par morceaux :

$$C^u_{k_u}\left(\left[\left(\frac{k_v}{n_v}\right), \left(\frac{k_v+1}{n_v}\right)\right]\right), k_u = 1, ..., n_u; k_v = 0, ..., n_v - 1 \tag{3.8a}$$

et

$$C^v_{k_v}\left(\left[\left(\frac{k_u}{n_u}\right), \left(\frac{k_u+1}{n_u}\right)\right]\right), k_v = 1, ..., n_v; k_u = 0, ..., n_u - 1 \tag{3.8b}$$

de sorte qu'elles conservent les tangentes aux points raccords et qu'elles passent par ces nouveaux points définis.

La modification d'un seul segment de courbe peut donc se décrire de la façon suivante : nous devons modifier le morceau de courbe de sorte qu'il ait deux points aux extrémités différents de ceux qu'il avait, mais il doit conserver les mêmes tangentes.

Nous devons donc résoudre le problème suivant : soit la courbe C définie par la fonction f sur $[a, b]$ de longueur L de sorte que :

$$\begin{aligned} f(a) &= P_0 \\ f(b) &= P_1 \\ f'(a) &= T_0 \\ f'(b) &= T_1 \end{aligned}$$

et nous cherchons une nouvelle courbe $\widetilde{C}$ proche de C définie par $\widetilde{f}$ sur $[a, b]$ qui doit satisfaire les conditions suivantes :

$$\begin{aligned} \widetilde{f}(a) &= \widetilde{P}_0 \\ \widetilde{f}(b) &= \widetilde{P}_1 \\ \widetilde{f}'(a) &= T_0 \\ \widetilde{f}'(b) &= T_1 \end{aligned}$$

de longueur $\widetilde{L}$ aussi proche que possible de L. Nous allons donc modifier la fonction de départ par la fonction de Hermite φ_1 (voir son graphe FIG.3.5) afin de modifier les valeurs des points extrêmes :

$$\widetilde{f}(t) = f(t) + \left(1 - \varphi_1\left(\frac{t-a}{b-a}\right)\right).\overrightarrow{D}_0 + \varphi_1\left(\frac{t-a}{b-a}\right).\overrightarrow{D_1} \tag{3.9}$$

avec :

$$\begin{aligned} \varphi_1(u) &= 3u^2 - 2u^3 \\ \overrightarrow{D_0} &= \widetilde{P}_0 - P_0 \\ \overrightarrow{D}_1 &= \widetilde{P}_1 - P_1 \end{aligned}$$

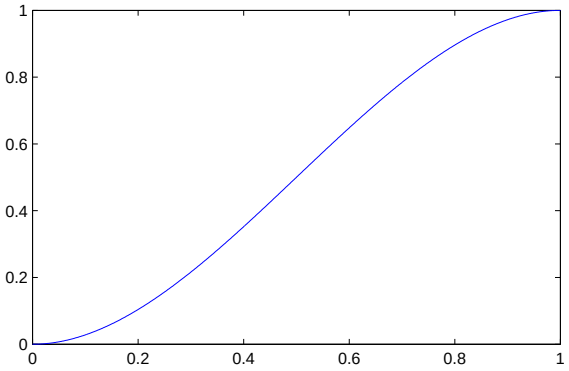


FIG. 3.5 – fonction de Hermite φ_1

Nous avons alors :

$$\widetilde{f}(a) = f(a) + \overrightarrow{D}_0 = \widetilde{P}_0 \tag{3.10a}$$
$$\widetilde{f}(b) = f(b) + \overrightarrow{D}_1 = \widetilde{P}_1 \tag{3.10b}$$
$$\widetilde{f}'(a) = f'(a) = T_0 \tag{3.10c}$$
$$\widetilde{f}'(b) = f'(b) = T_1 \tag{3.10d}$$

Nous pouvons comparer la longueur de la nouvelle courbe à celle de la courbe initiale :

$$\begin{aligned} \widetilde{L} &= \int_a^b \left\|\widetilde{f}'(t)\right\| dt = \int_a^b \left\|f'(t) + \frac{1}{b-a}\varphi_1'\left(\frac{t-a}{b-a}\right).\left(\overrightarrow{D_1} - \overrightarrow{D_0}\right)\right\| dt \\ &\leq \int_a^b \left\|f'(t)\right\| dt + \int_0^1 \left\|\varphi_1'(t).\left(\overrightarrow{D_1} - \overrightarrow{D_0}\right)\right\| dt \\ &\leq L + \left\|\overrightarrow{D_1} - \overrightarrow{D_0}\right\| \int_0^1 |\varphi_1'(t)|\ dt \\ \widetilde{L} - L &\leq \left\|\overrightarrow{D_1} - \overrightarrow{D_0}\right\| \end{aligned}$$

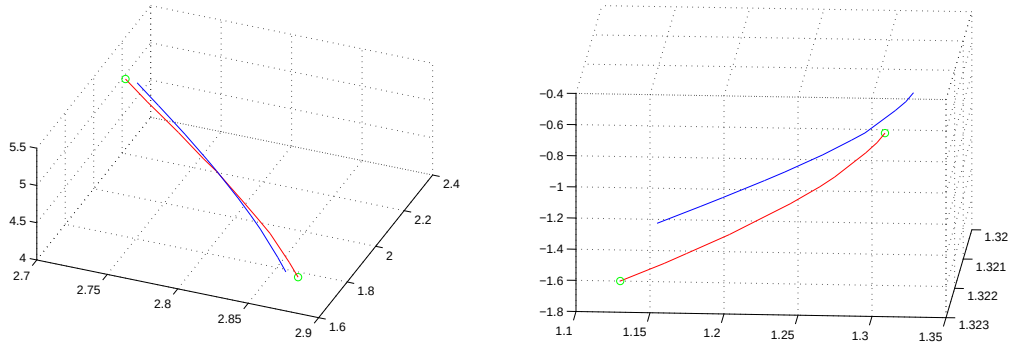
De même :

$$\begin{aligned} L &= \int_a^b \left\| f'(t) \right\| dt = \int_a^b \left\| \widetilde{f}'(t) - \frac{1}{b-a}\varphi_1'\left(\frac{t-a}{b-a}\right).\left(\overrightarrow{D_1} - \overrightarrow{D_0}\right) \right\| dt \\ L - \widetilde{L} &\leq \left\| \overrightarrow{D_1} - \overrightarrow{D_0} \right\| \end{aligned}$$

Nous obtenons ainsi :

$$\left| L - \widetilde{L} \right| \leq \left\| \overrightarrow{D_1} - \overrightarrow{D_0} \right\| \tag{3.11}$$

Nous obtenons une courbe qui satisfait les contraintes de positions et tangentes aux extrémités dont la longueur est d'autant plus proche de la courbe initiale que les déplacements du point de départ et de celui d'arrivée sont proches des points initiaux. Voyons sur les courbes de la figure FIG.3.6 deux morceaux de courbes obtenues avec cette méthode.



(a) Exemple 1 (b) Exemple 2

FIG. 3.6 – Modification de courbes en décalant les points de départ et d'arrivée, *(bleu) : courbes initiales, (rouge) : courbes modifiées, (vert) : nouveaux points à atteindre*

Le fait de définir tous les points d'intersection P_{k_u,k_v} dès le départ est un point faible de cette méthode. En effet, toutes nos courbes initiales sont calées à partir de leur seul point de départ, ainsi les erreurs de croisement (distance entre les 2 points sur les courbes orthogonales censés être identiques) sont plus élevées lorsque l'on s'éloigne du point de départ de la surface (nous pouvons voir cela comme une accumulation des erreurs). Il en résulte que les déplacements seront plus importants pour les points qui sont le plus loin du point de départ de la surface, donc les erreurs de longueur également (d'après la formule 3.11).

Nous allons définir une deuxième méthode de modification des courbes existantes pour recaler le réseau de départ, qui apportera une amélioration de la méthode précédente.

A.b. Deuxième méthode La méthode qui suit ne calcule pas tous les points d'intersection d'un coup, mais le fait progressivement : nous modifions le premier point d'intersection,

modifions les 2 courbes concernées entre le point de départ et ce nouveau point d'intersection, puis translatons le reste de ces deux courbes afin qu'elles restent continues (retrouver la continuité permet en fait de retrouver la régularité C^2), et nous itérons. Voyons ceci en détail dans ce qui suit.

Nous cherchons tout d'abord le point d'intersection pour $k_u = 1$ et $k_v = 1$:

$$P_{1,1} = \frac{1}{2}\left(C_1^u\left(\frac{1}{n_v}\right) + C_1^v\left(\frac{1}{n_u}\right)\right) \tag{3.12}$$

Nous devons modifier les courbes C_1^u sur $[0, \frac{1}{n_v}]$ et C_1^v sur $[0, \frac{1}{n_u}]$ de sorte qu'elles gardent les mêmes caractéristiques de tangentes aux extrémités, mais aient leur point d'arrivée modifié. Les courbes C_1^u et C_1^v doivent également être modifiées entre $[\frac{1}{n_v}, 1]$ et $[\frac{1}{n_u}, 1]$ pour qu'elles soient continues : nous les translatons du vecteur déplacement effectué.

Puis nous effectuons cette étape sur tous les autres morceaux des courbes.

Ainsi, nous devons construire une courbe à partir d'une courbe initiale, qui a au départ les mêmes données (position et tangente) mais aboutit à un point différent avec la même tangente à l'arrivée.

Nous devons résoudre le problème suivant : soit la courbe C définie par la fonction f sur $[a, b]$ de longueur L de sorte que :

$$\begin{aligned} f(a) &= P_0 \\ f(b) &= P_1 \\ f'(a) &= T_0 \\ f'(b) &= T_1 \end{aligned}$$

et nous cherchons une nouvelle courbe $\widetilde{C}$ proche de C définie par $\widetilde{f}$ sur $[a, b]$ qui doit satisfaire les conditions suivantes :

$$\begin{aligned} \widetilde{f}(a) &= P_0 \\ \widetilde{f}(b) &= \widetilde{P}_1 \\ \widetilde{f}'(a) &= T_0 \\ \widetilde{f}'(b) &= T_1 \end{aligned}$$

de longueur $\widetilde{L}$ aussi proche que possible de L.

Nous utilisons également la fonction de Hermite φ_1 :

$$\widetilde{f}(t) = f(t) + \varphi_1\left(\frac{t-a}{b-a}\right).\overrightarrow{D} \tag{3.13}$$

avec :

$$\overrightarrow{D} = \widetilde{P}_1 - P_1$$

Nous obtenons bien les contraintes requises :

$$\widetilde{f}(a) = f(a) = P_0 \tag{3.14a}$$
$$\widetilde{f}(b) = f(b) + \overrightarrow{D} = \widetilde{P}_1 \tag{3.14b}$$
$$\widetilde{f}'(a) = f'(a) = T_0 \tag{3.14c}$$
$$\widetilde{f}'(b) = f'(b) = T_1 \tag{3.14d}$$

Nous pouvons comparer $\widetilde{L}$ avec L :

$$\begin{aligned}
\widetilde{L} &= \int_a^b \left\|\widetilde{f}'(t)\right\| dt = \int_a^b \left\|f'(t) + \frac{1}{b-a}\varphi_1'\left(\frac{t-a}{b-a}\right).\overrightarrow{D}\right\| dt \\
&\leq \int_a^b \left\|f'(t)\right\| dt + \int_0^1 \left\|\varphi_1'(t).\overrightarrow{D}\right\| dt \leq L + \left\|\overrightarrow{D}\right\| \\
\widetilde{L} - L &\leq \left\|\overrightarrow{D}\right\|
\end{aligned}$$

De même :

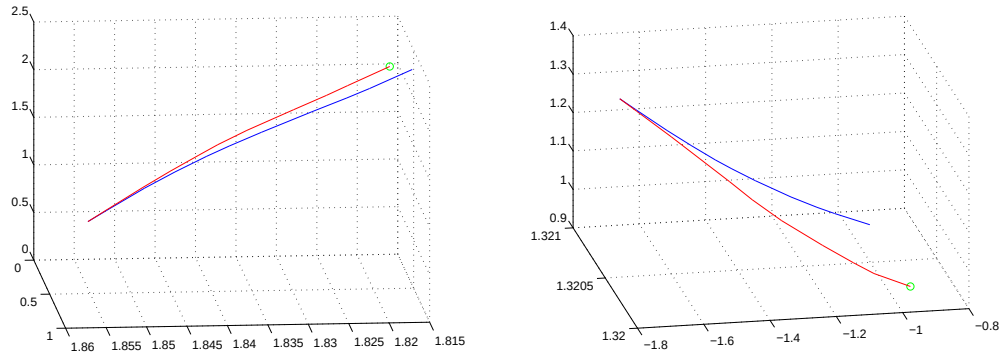
$$\begin{aligned}
L &= \int_a^b \left\|f'(t)\right\| dt = \int_a^b \left\|\widetilde{f}'(t) - \frac{1}{b-a}\varphi_1'\left(\frac{t-a}{b-a}\right).\overrightarrow{D}\right\| dt \\
L - \widetilde{L} &\leq \left\|\overrightarrow{D}\right\|
\end{aligned}$$

Nous avons finalement :

$$\left|L - \widetilde{L}\right| \leq \left\|\overrightarrow{D}\right\| \tag{3.15}$$

Nous obtenons une courbe qui satisfait les contraintes de positions et tangentes aux extrémités dont la longueur est d'autant plus proche de la courbe initiale que le déplacement du point d'arrivée est proche de l'initial.

Voyons également deux résultats obtenus à partir de cette deuxième façon de faire (voir FIG.3.7).



(a) Exemple 1 (b) Exemple 2

FIG. 3.7 – Modification de courbes en décalant les points d'arrivée seulement, *(bleu) : courbes initiales, (rouge) : courbes modifiées, (vert) : nouveaux points à atteindre*

Nous venons de voir deux méthodes qui permettent de fermer un réseau de courbes en choisissant des points d'intersection et qui modifient les courbes initiales pour qu'elles passent par ces points-là. Nous venons de voir l'influence de ces modifications sur les longueurs : leur importance dépend des déplacements des points de passage (que nous pouvons considérer assez faibles car les courbes initiales sont obtenues à partir de la méthode développée dans le chapitre 1, que nous avons validée).

Cependant, une autre méthode est envisagée si nous considérons la contrainte de longueur comme prioritaire dans certaines résolutions. Nous allons chercher les points d'intersection de sorte que la contrainte de longueur soit respectée : c'est ce que nous voyons dans la suite.

B. RECHERCHE DE NOUVELLES COURBES SATISFAISANT LES CONTRAINTES DE LONGUEUR

Dans les méthodes précédentes, nous ajoutions à nos courbes une quantité faible qui permettait de modifier les points et tangentes, en modifiant très peu les longueurs. Ici nous allons chercher à exprimer une nouvelle courbe proche de l'ancienne, avec des points de passage et des tangentes facilement identifiables dans la modélisation et où nous imposerions la longueur : nous allons utiliser la décomposition en fonctions cubiques de Hermite. Nous allons chercher un nouveau réseau de courbes, proche de l'ancien, composé de courbes cubiques par morceaux décrites sur la base des fonctions de Hermite.

Voici l'algorithme utilisé :

pour k_u de 1 à n_u
  pour k_v de 1 à n_v
    modifier les courbes $C^u_{k_u}$ sur $\left[\frac{k_v-1}{n_v}, \frac{k_v}{n_v}\right]$ et $C^v_{k_v}$ sur $\left[\frac{k_u-1}{n_u}, \frac{k_u}{n_u}\right]$ en déterminant leur nouveau point d'arrivée commun P_{k_u,k_v} tel que leur tangente aux extrémités, leur longueur et leur point de départ soient conservés,
    translater la courbe $C^u_{k_u}$ entre $\frac{k_v}{n_v}$ et 1 pour qu'elle reste continue (idem pour $C^v_{k_v}$ entre $\frac{k_u}{n_u}$ et 1)

Le problème élémentaire est donc le suivant : soit la courbe C_1 définie entre a_1 et b_1, de sorte que :

$$\begin{aligned} C_1(a_1) &= P_1 \text{ fixé}, \\ C_1'(a_1) &= T_{1,0} \text{ fixé}, \\ C_1'(b_1) &= T_{1,1} \text{ fixé}. \end{aligned}$$

De même pour la courbe C_2 définie entre a_2 et b_2 avec

$$\begin{aligned} C_2(a_2) &= P_2 \text{ fixé}, \\ C_2'(a_2) &= T_{2,0} \text{ fixé}, \\ C_2'(b_2) &= T_{2,1} \text{ fixé}. \end{aligned}$$

Nous cherchons leur point d'arrivée commun $Q = C_2(b_2) = C_1(b_1)$ telle que la longueur de la courbe C_1 soit L_1 et la courbe C_2 soit de longueur L_2. Ce point commun doit être proche des anciens points d'arrivée des courbes initiales ($Q_1 = (x_{q_1}, y_{q_1}, z_{q_1})$ pour C_1 et $Q_2 = (x_{q_2}, y_{q_2}, z_{q_2})$ pour C_2).

Nous cherchons les courbes sous forme de Hermite (voir les courbes de Hermite FIG.2.1

page 86) :

$$\begin{aligned} C_1(t) &= P_1.\varphi_0\left(\frac{t-a_1}{h_1}\right) + Q.\varphi_1\left(\frac{t-a_1}{h_1}\right) && (3.16a)\\ &+ h_1T_{1,0}.\varphi_2\left(\frac{t-a_1}{h_1}\right) + h_1T_{1,1}.\varphi_3\left(\frac{t-a_1}{h_1}\right), a_1 < t < b_1 \\ C_2(v) &= P_2.\varphi_0\left(\frac{v-a_2}{h_2}\right) + Q.\varphi_1\left(\frac{v-a_2}{h_2}\right) && (3.16b)\\ &+ h_2T_{2,0}.\varphi_2\left(\frac{v-a_2}{h_2}\right) + h_2T_{2,1}.\varphi_3\left(\frac{v-a_2}{h_2}\right), a_2 < v < b_2 \end{aligned}$$

avec

$$\begin{aligned} h_1 &= b_1 - a_1 \\ h_2 &= b_2 - a_2. \end{aligned}$$

Dans la formulation, l'inconnue est donc le point Q que l'on peut écrire par la suite sous forme de trois inconnues scalaires : $Q = (x, y, z)$.

Regardons les contraintes de longueur (calculs déjà vus lors de la déformation de courbes du chapitre 2 en page 81) :

$$1 = S_{simp}\left(\sqrt{\begin{array}{c} A_1 + B_{1,1}(x - x_{q_1}) + B_{1,2}(y - y_{q_1}) + B_{1,3}(z - z_{q_1}) \\ +\frac{\varphi_1'^2}{h_1^2}\left((x - x_{q_1})^2 + (y - y_{q_1})^2 + (z - z_{q_1})^2\right) \end{array}}\right) \qquad (3.17)$$

pour la courbe 1 avec

$$\begin{aligned} A_1 &= \frac{\varphi_0'}{h_1}\left(\frac{\varphi_0'}{h_1}\|P_1\|^2 + 2\frac{\varphi_1'}{h_1}\langle P_1, Q_1\rangle + 2\varphi_2'\langle P_1, T_{1,0}\rangle + 2\varphi_3'\langle P_1, T_{1,1}\rangle\right) && (3.18a)\\ &+\frac{\varphi_1'}{h_1}\left(\frac{\varphi_1'}{h_1}\|Q_1\|^2 + 2\varphi_2'\langle Q_1, T_{1,0}\rangle + 2\varphi_3'\langle Q_1, T_{1,1}\rangle\right) \\ &+\varphi_2'^2\|T_{1,0}\|^2 + 2\varphi_2'\varphi_3'\langle T_{1,0}, T_{1,1}\rangle \\ &+\varphi_3'^2\|T_{1,1}\|^2 \end{aligned}$$

$$B_{1,1} = 2\frac{\varphi_1'}{h_1}\left(\frac{\varphi_0'}{h_1}P_{1,x} + \frac{\varphi_1'}{h_1}x_{q_1} + \varphi_2'T_{1,0,x} + \varphi_3'T_{1,1,x}\right) \qquad (3.18b)$$

$$B_{1,2} = 2\frac{\varphi_1'}{h_1}\left(\frac{\varphi_0'}{h_1}P_{1,y} + \frac{\varphi_1'}{h_1}y_{q_1} + \varphi_2'T_{1,0,y} + \varphi_3'T_{1,1,y}\right) \qquad (3.18c)$$

$$B_{1,3} = 2\frac{\varphi_1'}{h_1}\left(\frac{\varphi_0'}{h_1}P_{1,z} + \frac{\varphi_1'}{h_1}z_{q_1} + \varphi_2'T_{1,0,z} + \varphi_3'T_{1,1,z}\right) \qquad (3.18d)$$

et

$$S_{simp}(v) = \frac{1}{3N}\left(v_1 + v_{N+1} + 4\sum_{\substack{i=2\\ i \text{ pair}}}^{N} v_i + 2\sum_{\substack{i=3\\ i \text{ impair}}}^{N} v_i\right) \qquad (3.19)$$

La même contrainte apparait pour la courbe C_2.

Dans les deux contraintes, les inconnues apparaissent dans une somme finie de racines carrées où elles sont sous forme quadratique. Il n'existe pas de solution unique. Nous allons donc choisir la solution qui minimise l'énergie. Nous allons ainsi chercher à minimiser la quantité suivante :

$$\min\left(\left(\int_{a_1}^{b_1}\left\|C_1''\right\|^2\right)^2+\left(\int_{a_2}^{b_2}\left\|C_2''\right\|^2\right)^2\right) \tag{3.20}$$

Après calculs, nous obtenons :

$$\begin{aligned}\int_{a_1}^{b_1}\left\|C_1''\right\|^2 = h_1\int_0^1 \quad & \Big(\alpha_1+\beta_{1,1}\left(x-x_{q_1}\right)+\beta_{1,2}\left(y-y_{q_1}\right)+\beta_{1,3}\left(z-z_{q_1}\right)\\ & +\frac{\varphi_1''^2}{h_1^4}\left(\left(x-x_{q_1}\right)^2+\left(y-y_{q_1}\right)^2+\left(z-z_{q_1}\right)^2\right)\Big)\end{aligned} \tag{3.21}$$

avec

$$\begin{aligned}\alpha_1 &= = \frac{\varphi_0''}{h_1^3}\left(\frac{\varphi_0''}{h_1}\left\|P_1\right\|^2+2\frac{\varphi_1''}{h_1}\left\langle P_1,Q_1\right\rangle+2\varphi_2''\left\langle P_1,T_{1,0}\right\rangle+2\varphi_3''\left\langle P_1,T_{1,1}\right\rangle\right) \quad (3.22a)\\ &+\frac{\varphi_1''}{h_1^3}\left(\frac{\varphi_1''}{h_1}\left\|Q_1\right\|^2+2\varphi_2''\left\langle Q_1,T_{1,0}\right\rangle+2\varphi_3''\left\langle Q_1,T_{1,1}\right\rangle\right)\\ &+\frac{1}{h_1^2}\left(\varphi_2''^2\left\|T_{1,0}\right\|^2+2\varphi_2''\varphi_3''\left\langle T_{1,0},T_{1,1}\right\rangle\right)\\ &+\frac{\varphi_3''^2}{h_1^2}\left\|T_{1,1}\right\|^2\\ \beta_{1,1} &= 2\frac{\varphi_1''}{h_1^3}\left(\frac{\varphi_0''}{h_1}P_{1,x}+\frac{\varphi_1''}{h_1}x_{q_1}+\varphi_2''T_{1,0,x}+\varphi_3''T_{1,1,x}\right) \quad (3.22b)\\ \beta_{1,2} &= 2\frac{\varphi_1''}{h_1^3}\left(\frac{\varphi_0''}{h_1}P_{1,y}+\frac{\varphi_1''}{h_1}y_{q_1}+\varphi_2''T_{1,0,y}+\varphi_3''T_{1,1,y}\right) \quad (3.22c)\\ \beta_{1,3} &= 2\frac{\varphi_1''}{h_1^3}\left(\frac{\varphi_0''}{h_1}P_{1,z}+\frac{\varphi_1''}{h_1}z_{q_1}+\varphi_2''T_{1,0,z}+\varphi_3''T_{1,1,z}\right) \quad (3.22d)\end{aligned}$$

En résumé, nous effectuons une minimisation d'énergie sous 2 contraintes de longueur :

$$\min\left(\left(\int_{a_1}^{b_1}\left\|C_1''\right\|^2\right)^2+\left(\int_{a_2}^{b_2}\left\|C_2''\right\|^2\right)^2\right) \tag{3.23}$$

$$\begin{cases} 1 = S_{simp}\left(\sqrt{\begin{array}{c}A_1+B_{1,1}\left(x-x_{q_1}\right)+B_{1,2}\left(y-y_{q_1}\right)+B_{1,3}\left(z-z_{q_1}\right)\\ +\frac{\varphi_1'^2}{h_1^2}\left(\left(x-x_{q_1}\right)^2+\left(y-y_{q_1}\right)^2+\left(z-z_{q_1}\right)^2\right)\end{array}}\right)\\ 1 = S_{simp}\left(\sqrt{\begin{array}{c}A_2+B_{2,1}\left(x-x_{q_2}\right)+B_{2,2}\left(y-y_{q_2}\right)+B_{2,3}\left(z-z_{q_2}\right)\\ +\frac{\varphi_1'^2}{h_2^2}\left(\left(x-x_{q_2}\right)^2+\left(y-y_{q_2}\right)^2+\left(z-z_{q_2}\right)^2\right)\end{array}}\right)\end{cases}$$

Nous obtenons ainsi le point d'intersection $Q=(x,y,z)$. D'où l'expression des deux morceaux de courbes C_1 et C_2.

Nous obtenons ainsi par itération de cette méthode pour tous les points d'intersection un réseau de courbes fermé.

3.4 DEUXIÈME CAS : LES DONNÉES SONT DANS UNE SEULE DIRECTION

Voyons à présent comment nous pouvons reconstruire un réseau de courbes fermé à partir de données acquises avec une seule famille de courbes dans la même direction, c'est-à-dire que nous avons des contraintes de tangentes dans les deux directions, mais des contraintes de longueur dans une seule. Voici la façon dont nous allons procéder :

- Etape 1 : nous reconstruisons en premier lieu la courbe qui lie tous les points de départ de la famille de rubans instrumentés avec les contraintes de tangentes et longueur (courbe verte trait plein de la figure FIG.3.8),
- Etape 2 : nous allons ensuite reconstruire les courbes dans la direction contrainte à partir des données tangentes et longueur (courbes rouges de la figure),
- Etape 3 : enfin nous reconstruisons les courbes dans l'autre sens avec les données tangentielles et les données de positions nouvellement connues (courbes vertes en pointillé).

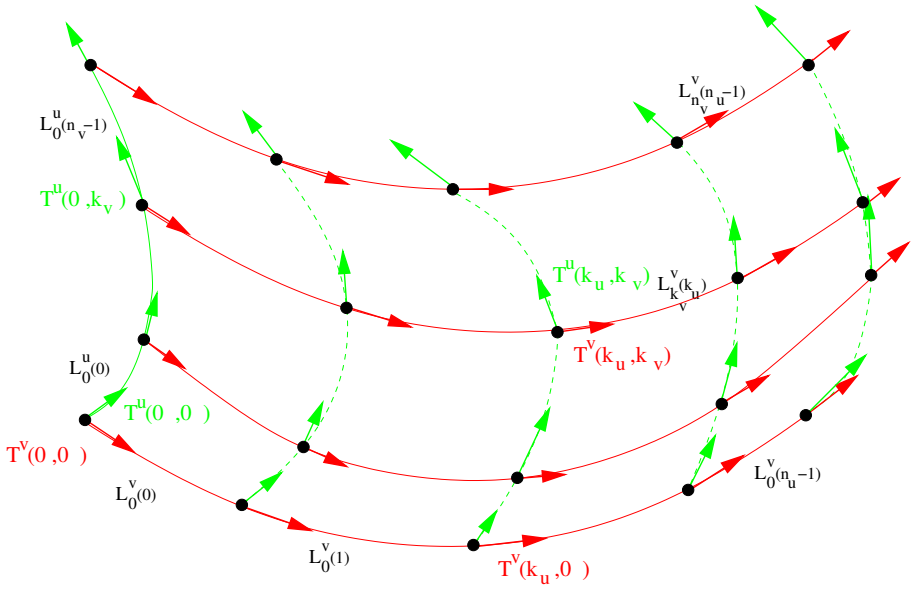


FIG. 3.8 – Etapes de reconstruction du réseau

3.4.1 CRÉATION DES COURBES AYANT DES CONTRAINTES DE TANGENTES ET DE LONGUEUR

ETAPE 1

Nous reconstruisons tout d'abord la courbe $C^u_0(v)$ avec les contraintes de répartition des capteurs

$$L^u_0(i), i = 0, ..., n_v - 1$$

et les données des tangentes

$$T_0^u(k_v), k_v = 0, ..., n_v.$$

Nous pouvons appliquer la méthode de reconstruction des courbes gauches vues dans le premier chapitre.

Nous connaissons alors les points de départ de toutes les courbes $\left(C_{k_v}^v(u), k_v = 0, ..., n_v\right)$:

$$C_{k_v}^v(0) = C_0^u\left(\frac{k_v}{n_v}\right), k_v = 0, ..., n_v.$$

ETAPE 2

Nous pouvons à présent reconstruire les courbes $\left(C_{k_v}^v(u), k_v = 0, ..., n_v\right)$ avec les contraintes de répartition des capteurs

$$L_{k_v}^v(k_u), k_v = 0, ..., n_v, k_u = 0, ..., n_u - 1$$

et les données tangentielles

$$T_{k_v}^v(k_u), k_u = 0, ..., n_u.$$

Ces courbes sont reconstruites avec la méthode que nous avons développée en première partie.

Ces courbes nous donnent les positions des capteurs :

$$P_{k_u,k_v} = C_{k_v}^v\left(\frac{k_u}{n_u}\right), k_u = 1, ..., n_u, k_v = 1, ..., n_v$$

qui vont nous permettre de reconstruire les courbes dans l'autre sens.

3.4.2 ETAPE 3 : RECALAGE DU RÉSEAU AVEC LA CRÉATION DES COURBES DANS L'AUTRE DIRECTION

Nous avons ainsi n_u courbes à reconstruire $\left(C_{k_u}^u(v), k_u = 1, ..., n_u\right)$ avec des contraintes de positions et tangentes. Les tangentes sont données par les capteurs, et les points de passage sont donnés par la construction des courbes précédentes :

$$C_{k_u}^u\left(\frac{k_v}{n_v}\right) = P_{k_u,k_v}, k_u = 1, ..., n_u, k_v = 0, ..., n_v$$

Décrivons le problème de façon générale :

Soit une courbe $C(t)$ à reconstruire pour $0 \leq t \leq 1$, sachant qu'elle a les contraintes de positions et tangentes en $n+1$ points de paramètre que nous choisirons régulièrement répartis $t_i = \frac{i}{n}$ $i = 0, ..., n$:

$$\begin{aligned} C(t_i) &= P_i, i = 0, ..., n \text{ connus} \\ C'(t_i) &= \lambda_i T_i, i = 0, ..., n, \ T_i \text{ unitaires connus et } \lambda_i \text{ inconnues} \end{aligned}$$

Nous allons utiliser une décomposition sur la base de Hermite un peu modifiée :

$$C(t) = P_i\varphi_0\left(nt - i\right) + P_{i+1}\varphi_1\left(nt - i\right) + \lambda_i T_i\varphi_2\left(nt - i\right) + \lambda_{i+1}T_{i+1}\varphi_3\left(nt - i\right)$$

pour t entre $\frac{i}{n}$ et $\frac{i+1}{n}, i = 0, ..., n-1$.

Nous avons donc $n+1$ inconnues, les $\lambda_i, i = 0, ..., n$, que nous allons choisir de sorte à minimiser l'énergie de la courbe : $\min \int \|C''\|^2$.

$$\begin{aligned}\int_0^1 \|C''(t)\|^2 dt &= \sum_{i=0}^{n-1} \int_{\frac{i}{n}}^{\frac{i+1}{n}} \|C''(t)\|^2 dt \\ &= \frac{1}{n} \sum_{i=0}^{n-1} \int_0^1 n^4 \left\|P_i\varphi_0''(u) + P_{i+1}\varphi_1''(u) + \lambda_i T_i \varphi_2''(u) + \lambda_{i+1} T_{i+1}\varphi_3''(u)\right\|^2 du\end{aligned}$$

En développant, nous devons minimiser la forme quadratique suivante :

$$\sum_{i=0}^{n-1} A_i\lambda_i + B_i\lambda_i^2 + C_i\lambda_i\lambda_{i+1} + D_i\lambda_{i+1}^2 + E_i\lambda_{i+1} + F_i$$

$$\begin{aligned}\text{avec } A_i &= 2\langle P_i, T_i\rangle \int \varphi_0''\varphi_2'' + 2\langle P_{i+1}, T_i\rangle \int \varphi_1''\varphi_2'' \\ B_i &= \int \varphi_2''^2 \\ C_i &= 2\langle T_i, T_{i+1}\rangle \int \varphi_2''\varphi_3'' \\ D_i &= \int \varphi_3''^2 \\ E_i &= 2\langle P_i, T_{i+1}\rangle \int \varphi_0''\varphi_3'' + 2\langle P_{i+1}, T_{i+1}\rangle \int \varphi_1''\varphi_3'' \\ F_i &= \|P_i\|^2 \int \varphi_0''^2 + \|P_{i+1}\|^2 \int \varphi_1''^2 + 2\langle P_i, P_{i+1}\rangle \int \varphi_0''\varphi_1''\end{aligned}$$

Nous devons minimiser une fonction quadratique, donc chercher à annuler son gradient :

$$\nabla\left(\int \|C''\|^2\right) = \begin{pmatrix} A_0 + 2B_0\lambda_0 + C_0\lambda_1 \\ C_0\lambda_0 + E_0 + 2D_0\lambda_1 + A_1 + 2B_1\lambda_1 + C_1\lambda_2 \\ \vdots \\ C_{n-2}\lambda_{n-2} + E_{n-2} + 2D_{n-2}\lambda_{n-1} + A_{n-1} + 2B_{n-1}\lambda_{n-1} + C_{n-1}\lambda_n \\ C_{n-1}\lambda_n + 2D_{n-1}\lambda_n + E_{n-1}\end{pmatrix}$$

Nous obtenons un système linéaire. Nous cherchons le vecteur $X = (\lambda_0, \lambda_1, ..., \lambda_n)$, solution de $MX = Y$, avec M tridiagonale symétrique :

$$M = \begin{pmatrix} 2B_0 & C_0 & & & \\ C_0 & 2B_1 + 2D_0 & C_1 & & \\ & C_1 & \ddots & \ddots & \\ & & \ddots & 2B_{n-1} + 2D_{n-2} & C_{n-1} \\ & & & C_{n-1} & 2D_{n-1}\end{pmatrix}, Y = \begin{pmatrix} -A_0 \\ -A_1 - E_0 \\ \vdots \\ -A_{n-1} - E_{n-2} \\ -E_{n-1}\end{pmatrix}$$

Le système est inversible : nous obtenons une solution unique de notre vecteur composé des λ_i, nous obtenons ainsi des courbes qui minimisent l'énergie.

Nous reconstruisons toutes les courbes dans la deuxième direction avec cette méthode. Nous obtenons ainsi un réseau recalé à partir duquel nous pouvons définir une surface solution.

3.5 REMPLISSAGE DE LA SURFACE

Le réseau ainsi déterminé nous permet de calculer $n_u.n_v$ "patchs" à remplir. Le morceau de surface $S\left([\frac{k_u}{n_u}, \frac{k_u+1}{n_u}] \times [\frac{k_v}{n_v}, \frac{k_v+1}{n_v}]\right)$ est créé en utilisant les quatre morceaux de courbes $C_u^{k_u}([\frac{k_v}{n_v}, \frac{k_v+1}{n_v}])$, $C_u^{k_u+1}([\frac{k_v}{n_v}, \frac{k_v+1}{n_v}])$, $C_v^{k_v}([\frac{k_u}{n_u}, \frac{k_u+1}{n_u}])$ et $C_v^{k_v+1}([\frac{k_u}{n_u}, \frac{k_u+1}{n_u}])$.

Nous allons utiliser la méthode de Coons (voir [COO64], [COO74], [FAR02a],[BAR82], [BAR77], et [GOR83]), méthode que nous détaillons ici.

Nous devons calculer la surface $S(u, v)$ qui a pour courbes de bord $C_N(u) = S(u, 0)$, $C_S(u) = S(u, 1)$, $C_O(v) = S(0, v)$, $C_E(v) = S(1, v)$, avec les paramètres u et v variant de 0 à 1. Les 4 points extrèmes sont notés P_{00}, P_{01}, P_{10}, et P_{11} :

$$C_N(0) = P_{00} = C_O(0) \tag{3.24a}$$
$$C_N(1) = P_{10} = C_E(0) \tag{3.24b}$$
$$C_S(0) = P_{01} = C_O(1) \tag{3.24c}$$
$$C_S(1) = P_{11} = C_E(1) \tag{3.24d}$$

Dans ce cas, nous allons utiliser des fonctions de mélange pour définir la surface à partir de ces courbes de bord :

$$S(u, v) = S_{NS}(u, v) + S_{OE}(u, v) - S_b(u, v) \tag{3.25a}$$

telle que

$$S_{NS}(u, v) = F_1^v(v)\ C_N(u) + F_2^v(v)\ C_S(u) \tag{3.25b}$$
$$S_{OE}(u, v) = F_1^u(u) C_O(v) + F_2^u(u)\ C_E(v) \tag{3.25c}$$
$$\begin{aligned} S_b(u, v) &= F_1^v(v)\,(F_1^u(u)\ P_{00} + F_2^u(u)\ \ P_{10}) \\ &+ F_2^v(v)\,(F_1^u(u)\ P_{01} + F_2^u(u)\ \ P_{11}) \end{aligned} \tag{3.25d}$$

Nous utilisons comme fonction de mélange 2 fonctions de Hermite :

$$\begin{aligned} F_1^u &= F_1^v; F_2^u = F_2^v \\ F_1^v(v) &= 1 - 3u^2 + 2u^3; F_2^v(v) = 3u^2 - 2u^3 \end{aligned}$$

La surface est dans ce cas G^1, ce procédé de Coons à partir des fonctions de Hermite est dit partiellement bicubique.

Nous obtenons à présent une surface qui satisfait les contraintes données par les systèmes de mesure, à savoir contenir des familles de courbes ayant des contraintes de tangentes et de longueurs entre les capteurs. Nous allons à présent voir les résultats obtenus sur différentes surfaces tests pour ces diverses méthodes.

3.6 RÉSULTATS ET COMPARAISONS

3.6.1 RECONSTRUCTION DES RÉSEAUX DE COURBES

La première étape importante pour valider cette méthode de reconstruction de surfaces est de juger de la qualité des réseaux de courbes reconstruits. Nous allons en premier lieu comparer les méthodes de reconstruction du réseau lorsque nous avons pour données un filet de capteurs.

A. UTILISATION D'UN FILET DE DONNÉES

Dans les tests qui suivent, nous comparons sur différents exemples les trois méthodes mises en place pour la reconstruction du réseau de courbes lorsque nous avons un filet de données : la première méthode étant celle qui fixe dès le départ tous les points d'intersection puis modifie les courbes par ajout de fonction de Hermite ; la deuxième méthode fixe également les points d'intersection avant de modifier les courbes mais le fait progressivement en translatant les morceaux petit à petit pour faire en sorte que les vecteurs déplacements qui interviennent en facteur de la fonction de Hermite soient plus faibles ; enfin la dernière méthode est une méthode qui cherche de nouvelles courbes proches des anciennes qui satisfont les contraintes de longueur entre chaque capteur.

Test 1 : portion de cône Nous cherchons dans ce premier test à reconstruire un morceau de cône avec un filet de capteurs de taille cinq par cinq. Le morceau de cône est défini par l'équation suivante :

$$\begin{aligned} \text{pour} (u,v) \quad &\in \quad [0,1] \times [0,1] \\ &\left\{ \begin{pmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{pmatrix} = \left(\frac{u+v-1}{2} \right) \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} + \left(\frac{u+v+1}{2} \right) \begin{pmatrix} \left(\frac{u-v}{2} \right) \\ \left(\frac{u-v}{2} \right)^2 \\ \left(\frac{u-v}{2} \right)^3 \end{pmatrix} \right. \end{aligned}$$

Nous voyons la surface ainsi que le filet de données sur la figure FIG.3.9.

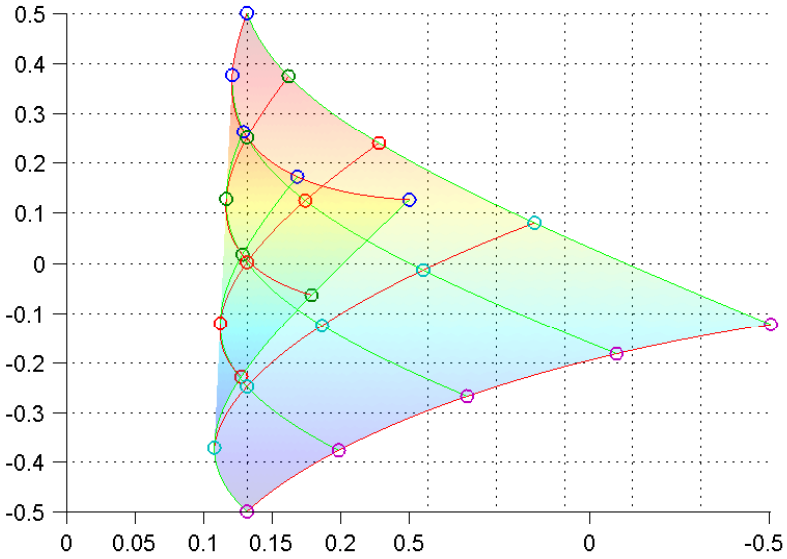
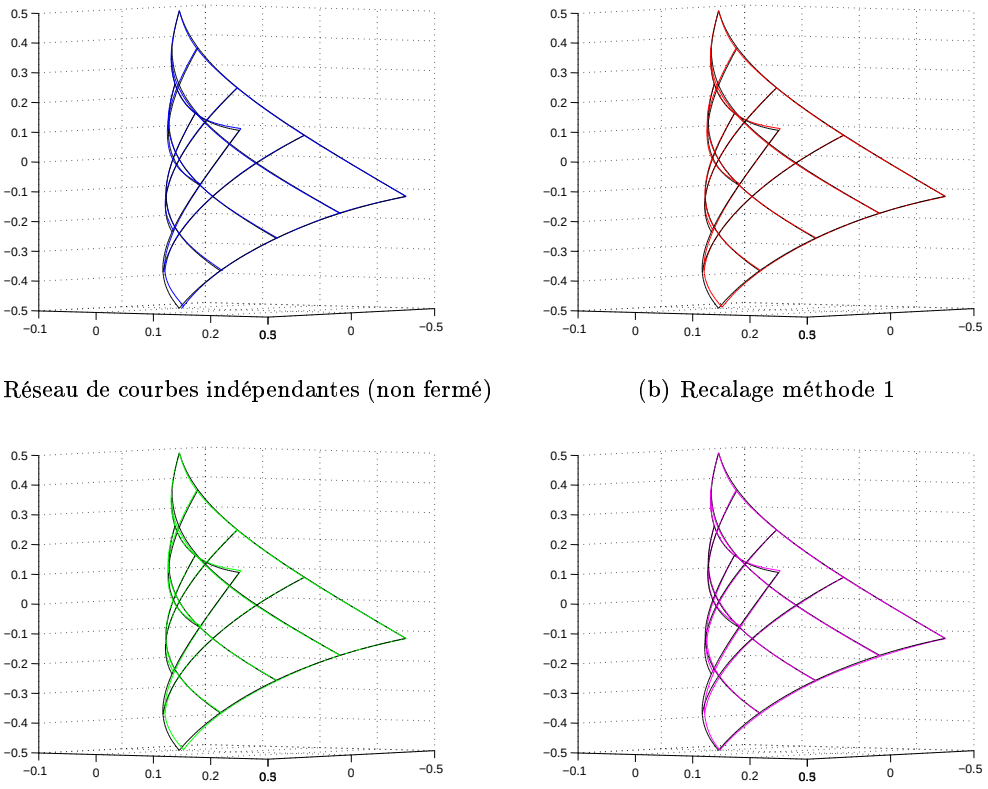


FIG. 3.9 – Portion de cône : réseau à reconstruire

Nous reconstruisons tout d'abord un réseau initial en créant les courbes indépendamment les unes des autres à partir de l'algorithme vu dans le premier chapitre. Nous le modifions alors avec l'une des trois méthodes dont nous avons parlé précédemment : nous pouvons comparer les résultats visuellement sur la figure FIG.3.10.



(a) Réseau de courbes indépendantes (non fermé)

(b) Recalage méthode 1

(c) Recalage méthode 2

(d) Recalage méthode 3

FIG. 3.10 – Portion de cône : Reconstructions du réseau de courbes

Visuellement, les trois méthodes donnent de bonnes reconstructions. Voyons à présent les différents calculs d'erreur que nous avons effectués sur les différents segments de courbes calculés.

Pour la modification de ce réseau, nous créons 4 morceaux de courbes pour chaque courbe initiale (car chaque courbe est connue grâce à 5 capteurs) et nous modifions 4 courbes dans chaque direction (les deux courbes de bord ne sont pas changées) : nous modifions alors 32 morceaux de courbes. Pour toutes ces modifications, nous calculons les erreurs suivantes :

- erreur de position du point d'arrivée (en pourcentage par rapport à la longueur de la courbe totale concernée),
- erreur de longueur par rapport à la longueur du morceau de courbe réelle,
- erreur en norme L^2 de la position totale du morceau de courbe reconstruit par rapport au morceau de courbe réel (normalisée par rapport à la longueur totale de la courbe concernée),
- erreur de Hausdorff du morceau de courbe (également normalisée par rapport à la longueur totale de la courbe concernée).

Voici donc les courbes d'erreur pour les 32 morceaux de courbes modifiés sur la figure FIG.3.11.

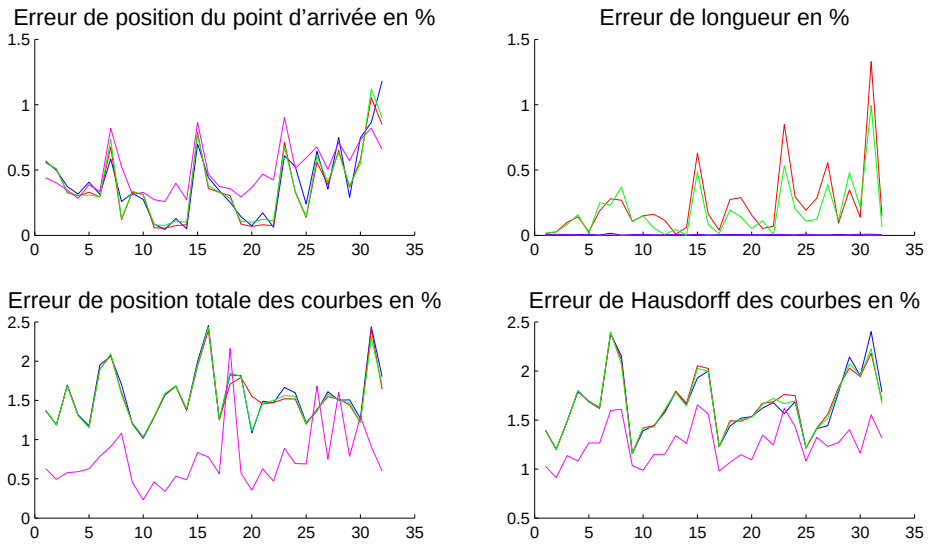


FIG. 3.11 – Erreurs de reconstruction du réseau de courbes pour la portion de cône, *(bleu) : erreurs du réseau de courbes indépendantes (non fermé), (rouge) : erreurs du réseau obtenu par la méthode 1, (vert) : méthode 2, (rose) : méthode de contraintes de longueur*

Nous pouvons voir plusieurs choses à partir de ces résultats. En premier lieu, les deux premières méthodes ont des erreurs très proches de celles de la méthode initiale (celle de reconstruction de courbes vue en première partie) concernant les positions des capteurs et les positions globales des courbes, ce sont de bons résultats, mais elles augmentent l'erreur de longueur (cette erreur reste tout de même assez faible, moins de 1.5%), et sont plus fortes lorsque nous sommes sur des morceaux de courbes plus éloignés du départ de la surface. Nous voyons aussi en général que ces erreurs de longueur sont légèrement plus faibles avec la deuxième méthode (courbes vertes), qui modifie progressivement les points d'intersection (méthode qui avait justement été conçue pour cette raison). En ce qui concerne la méthode de reconstruction de courbes avec contraintes de longueur, nous voyons que ses erreurs diffèrent des autres, car ce ne sont pas de légères modifications des courbes existantes, mais de nouvelles courbes. Nous voyons que les erreurs sont plus stables tout au long de la courbe, un peu plus élévées en ce qui concerne les positions des points d'intersection, mais donne pour cet exemple de meilleurs résultats pour la reconstruction globale du réseau. Les erreurs de longueur sont quant à elles très faibles, ce qui se justifie par le fait que les courbes sont construites à partir de cette contrainte.

Voyons maintenant un deuxième exemple afin de préciser ces résultats.

Test 2 : vagues Nous cherchons ici à reconstruire une surface à vagues avec un filet de capteurs de taille sept par sept. Elle a été obtenue avec la formulation suivante :

$$\text{pour}(u,v) \in [-4,4]\times[-4,4] \quad \begin{cases} x(u,v) = u \\ y(u,v) = v \\ z(u,v) = \sin(u+\sin(v)) \end{cases} \quad \text{puis rotation autour de } y \text{ de } \frac{\pi}{3}$$

Voyons sa représentation sur la figure FIG.3.12.

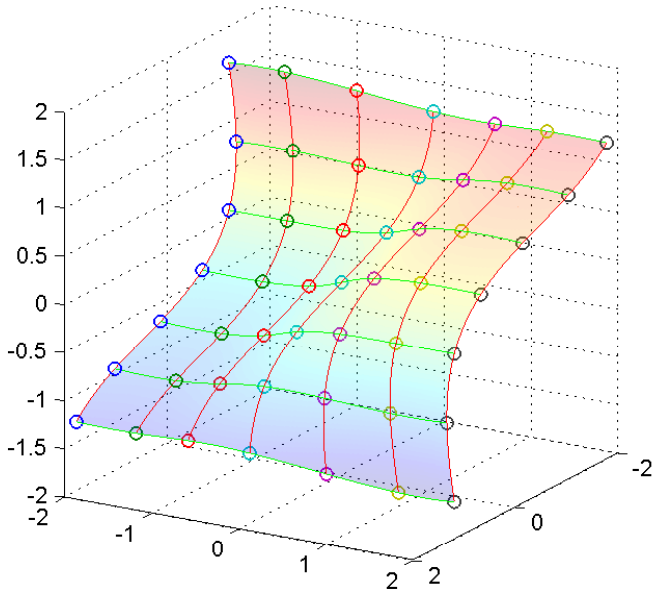
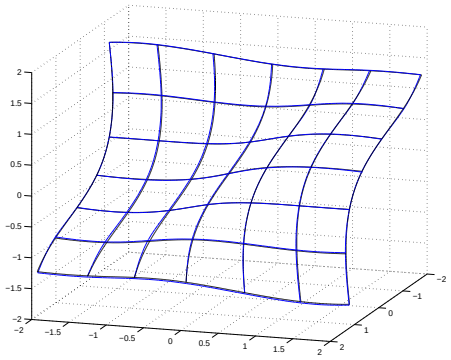
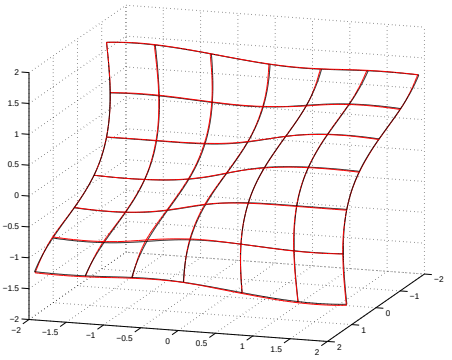


FIG. 3.12 – Vagues : réseau à reconstruire

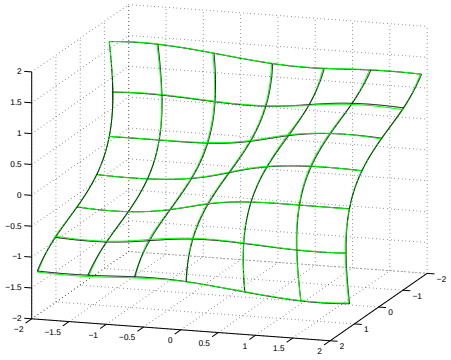
Voyons les différents réseaux obtenus avec les diverses méthodes sur la figure FIG.3.13.



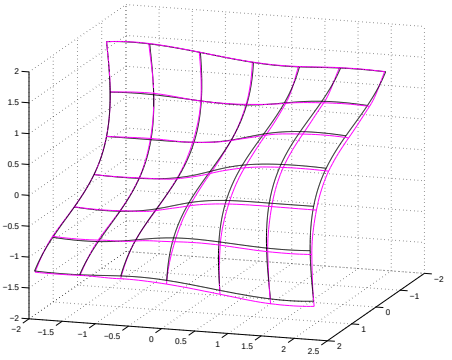
(a) Réseau de courbes indépendantes (non fermé)



(b) Recalage méthode 1



(c) Recalage méthode 2



(d) Recalage méthode 3

FIG. 3.13 – Vagues : Reconstructions du réseau de courbes

Le réseau initial de courbes indépendantes était visuellement très proche du réseau réel, il en découle clairement de très bons résultats des méthodes 1 et 2. La méthode 3 quant à elle donne un réseau légèrement plus éloigné.

Pour la modification de ce réseau de courbes, nous modifions alors $(7-1).(7-1).2$ morceaux de courbes, soient 72. Pour toutes ces modifications, nous calculons les mêmes erreurs que pour le test 1 ; les erreurs sont représentées dans la figure FIG.3.14.

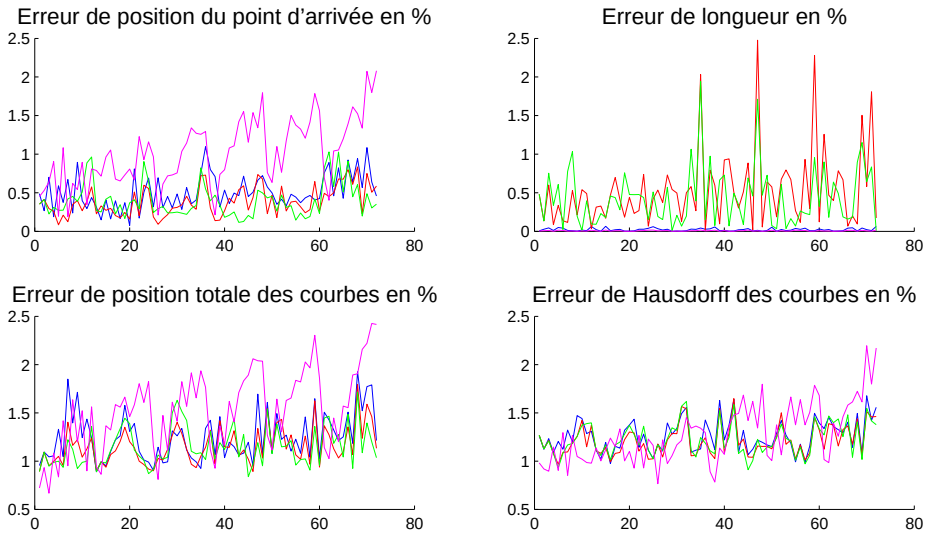


FIG. 3.14 – Erreurs de reconstruction du réseau de courbes pour les vagues, *(bleu) : erreurs du réseau de courbes indépendantes (non fermé), (rouge) : erreurs du réseau obtenu par la méthode 1, (vert) : méthode 2, (rose) : méthode de contraintes de longueur*

Nous voyons également à partir de ces résultats que les deux méthodes qui modifient seulement les courbes initiales ont leurs erreurs proches et en général plus faibles que les erreurs obtenues avec le réseau initial à fermer : il est en effet logique de voir que le fait de fermer le réseau amoindrit les erreurs de reconstruction. La deuxième méthode donne en général de meilleurs résultats que la première. En ce qui concerne la méthode de reconstruction avec contraintes de longueur, elle donne de très bons résultats pour la longueur des courbes obtenues, mais donne de moins bons résultats que les deux autres méthodes dans cet exemple.

Voyons les résultats sur un troisième exemple.

Test 3 : selle Nous cherchons ici à reconstruire une surface paraboloïde hyperbolique, son équation est la suivante :

$$\text{pour}(u,v) \in [-5,5]\times[-8,8] \quad \begin{cases} x(u,v) = 4u \\ y(u,v) = 3v \\ z(u,v) = u^2 - v^2 \end{cases}$$

On munit la surface d'une grille de 8 capteurs par 8 (voir FIG.3.15).

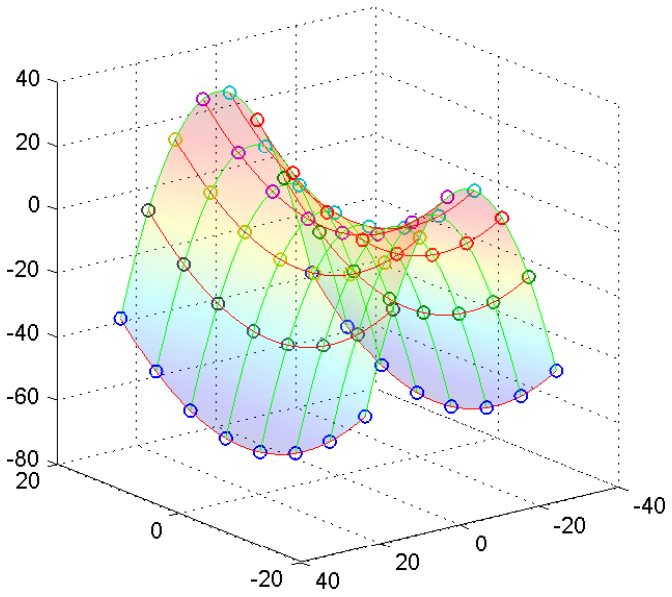
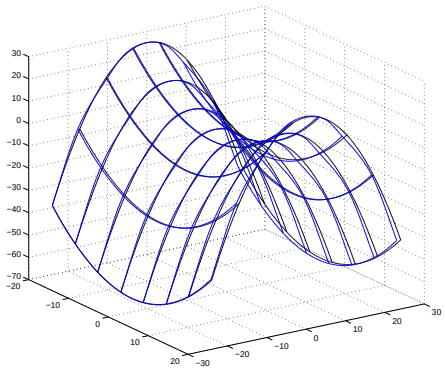
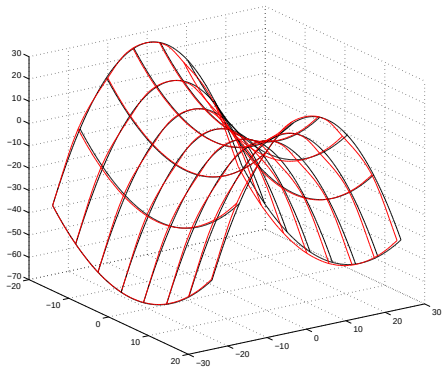


FIG. 3.15 – Selle : réseau à reconstruire

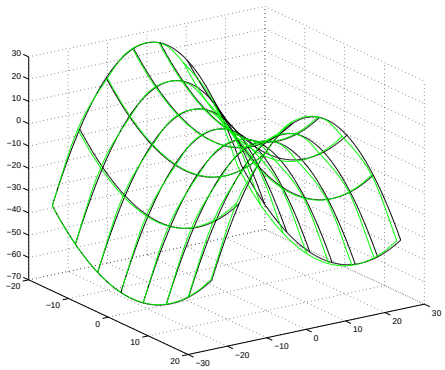
Les différents réseaux obtenus sont sur la figure FIG.3.16.



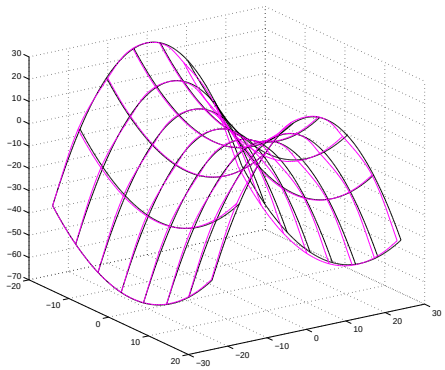
(a) Réseau de courbes indépendantes (non fermé)



(b) Recalage méthode 1



(c) Recalage méthode 2



(d) Recalage méthode 3

FIG. 3.16 – Selle : Reconstructions du réseau de courbes

Les résultats obtenus sont très proches visuellement du réseau réel. Voyons les erreurs calculées sur chaque morceau de courbes modifié (soit 98). Les erreurs sont rassemblées dans la figure FIG.3.17.

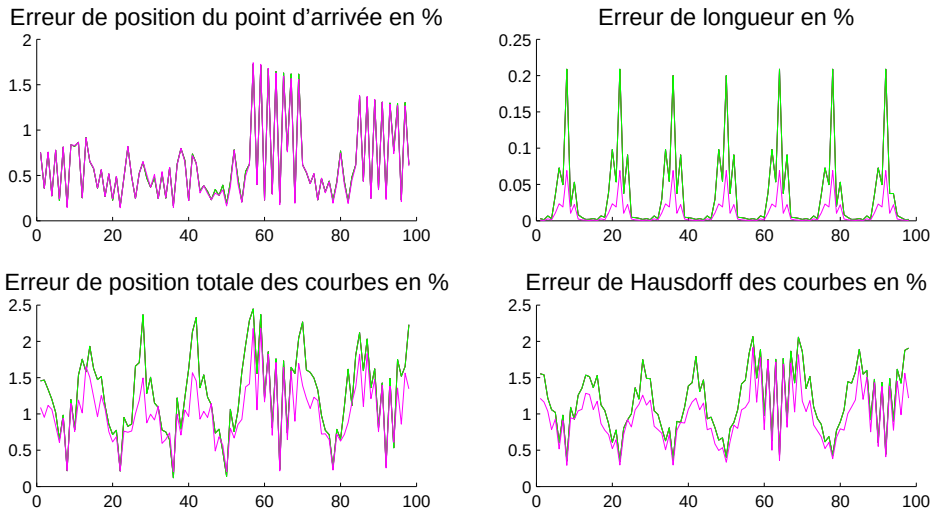


FIG. 3.17 – Erreurs de reconstruction du réseau de courbes pour la selle, *(bleu) : erreurs du réseau de courbes indépendantes (non fermé), (rouge) : erreurs du réseau obtenu par la méthode 1, (vert) : méthode 2, (rose) : méthode de contraintes de longueur*

Sur ces résultats, les courbes bleues, rouges et vertes sont confondues (nous ne voyons que les vertes), cela signifie que le réseau initial de courbes indépendantes était très proche du réseau réel, donc constituait presque un réseau fermé. Les modifications apportées furent alors très faibles, n'engendrant que très peu de différences dans les erreurs calculées. La méthode par contraintes de longueur donne également des résultats très proches de ceux du réseau initial. Cela se comprend également car c'est une méthode de minimisation qui prend comme données initiales les données du réseau initial, elle a donc convergé très rapidement en des points proches des initiaux, obtenant ainsi des résultats similaires aux autres (voire légèrement meilleurs en ce qui concerne le positionnement global de la courbe).

Test 4 : surface de Bézier Nous cherchons ici à reconstruire une surface de Bézier. Cette surface a été obtenue à partir de 16 points de contrôle et des polynômes de Bernstein de degré 3. Les points de contrôle sont les suivants :

$$\left\{\begin{array}{cccc} P_{00}=\begin{pmatrix}1\\1\\-2\end{pmatrix} & P_{01}=\begin{pmatrix}1\\0\\-1\end{pmatrix} & P_{02}=\begin{pmatrix}2\\0\\0\end{pmatrix} & P_{03}=\begin{pmatrix}3\\0\\4\end{pmatrix} \\ P_{10}=\begin{pmatrix}0\\1\\1\end{pmatrix} & P_{11}=\begin{pmatrix}1\\1\\2\end{pmatrix} & P_{12}=\begin{pmatrix}2\\1\\7\end{pmatrix} & P_{13}=\begin{pmatrix}3\\1\\4\end{pmatrix} \\ P_{20}=\begin{pmatrix}0\\2\\3\end{pmatrix} & P_{21}=\begin{pmatrix}1\\2\\8\end{pmatrix} & P_{22}=\begin{pmatrix}2\\2\\5\end{pmatrix} & P_{23}=\begin{pmatrix}3\\2\\3\end{pmatrix} \\ P_{30}=\begin{pmatrix}0\\3\\2\end{pmatrix} & P_{31}=\begin{pmatrix}1\\3\\1\end{pmatrix} & P_{32}=\begin{pmatrix}2\\3\\-1\end{pmatrix} & P_{33}=\begin{pmatrix}2.5\\2.5\\7\end{pmatrix} \end{array}\right\}$$

Cette surface est munie d'un réseau de 6 capteurs par 6 (voir FIG.3.18).

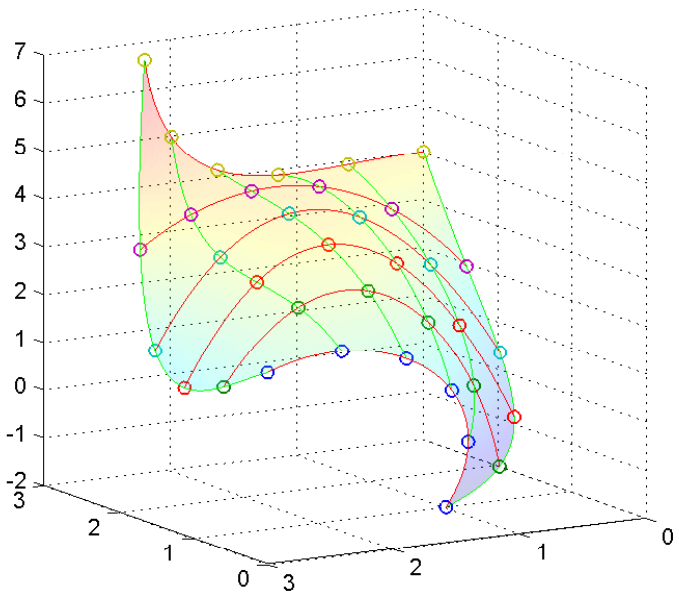
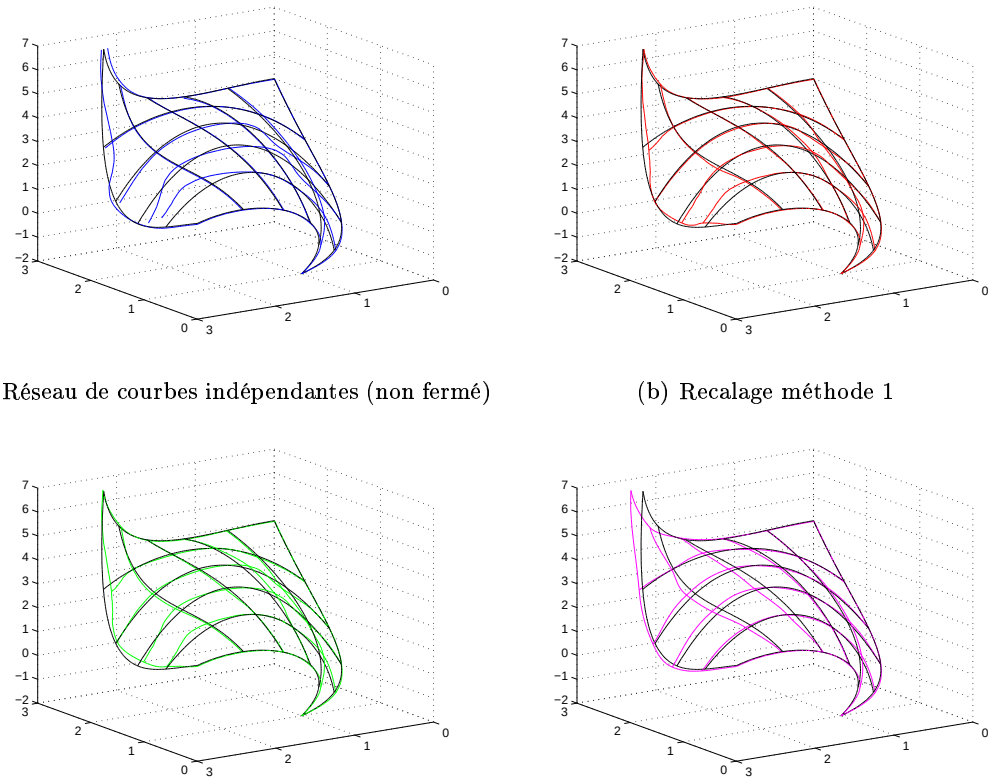


FIG. 3.18 – Surface de Bézier : réseau à reconstruire

Les différents réseaux obtenus sont sur la figure FIG.3.19.



(a) Réseau de courbes indépendantes (non fermé)

(b) Recalage méthode 1

(c) Recalage méthode 2

(d) Recalage méthode 3

FIG. 3.19 – Surface de Bézier : Reconstructions du réseau de courbes (point de départ $P_{00} = (1, 1, -2)$)

Visuellement, nous voyons que le réseau initial est assez mauvais, les points censés s'intersecter sont vraiment loin les uns des autres. Dans les trois réseaux modifiés, nous voyons que le fait de forcer les intersections améliorent le rendu visuel, avec une préférence pour la dernière méthode qui donne des courbes plus lisses. En effet, les courbes des méthodes 1 et 2 étant des modifications des courbes existantes, le fait que l'on fasse des déplacements importants pour les points d'intersection (du fait que le réseau initial ne soit pas satisfaisant) donne des courbes assez perturbées, contrairement aux courbes de la dernière méthode qui crée des courbes lisses par modélisation (splines cubiques de Hermite). Voyons les erreurs des courbes par morceaux ($(6-1).(6-1).2$ morceaux de courbes, soient 50) sur la figure FIG.3.20.

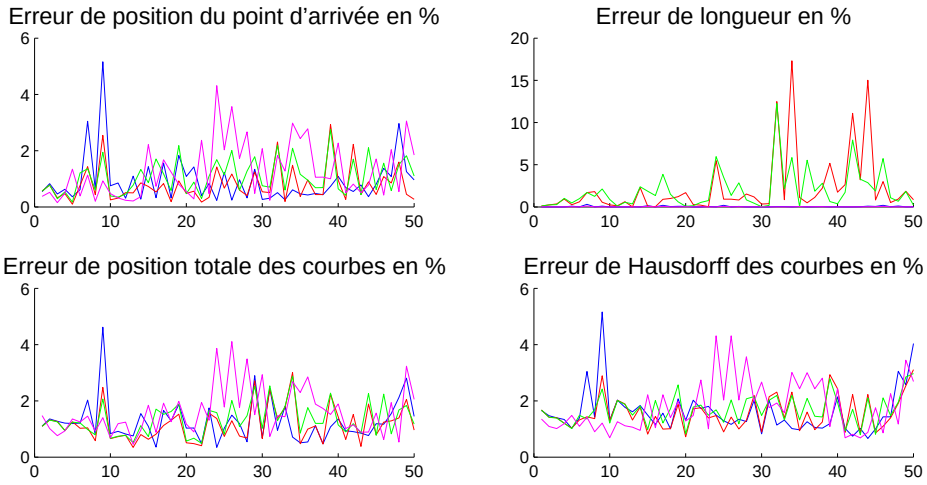


FIG. 3.20 – Erreurs de reconstruction du réseau de courbes pour la surface de Bézier, *(bleu) : erreurs du réseau initial, (rouge) : erreurs du réseau obtenu par la méthode 1, (vert) : méthode 2, (rose) : méthode de contraintes de longueur*

Ces erreurs confirment ce que nous voyions dans les graphes des réseaux. Tout d'abord, les méthodes 1 et 2 donnent des erreurs de longueur assez élevées pour quelques morceaux de courbes : cela s'explique du fait du mauvais réseau initial, qui force ainsi de forts déplacements des positions des points de mesure. En ce qui concerne les erreurs globales de position, elles donnent toutes des erreurs assez faibles en moyenne avec des pics pour les morceaux de courbes qui s'éloignent du point de départ de la surface. Les courbes restent proches du réseau initial pour deux raisons différentes selon la méthode employée : le fait d'une part d'être des modifications de ce réseau pour les méthodes 1 et 2, et le fait de contraindre les longueurs d'autre part pour le réseau obtenu par la méthode 3.

Conclusion Ces différents tests nous ont permis de tirer quelques conclusions sur ces méthodes. Les réseaux obtenus par les méthodes 1 et 2, qui modifient les courbes existantes en modulant avec une fonction de Hermite, peuvent provoquer de fortes erreurs des longueurs des courbes, comme le montrait les majorations d'erreur. Cependant, nous obtenons de meilleurs résultats en général en modifiant les points d'intersection progressivement plutôt que de les fixer dès le départ (la méthode 2 améliore les résultats par rapport à la méthode 1). De plus, nous pouvons dire que si le réseau initial donne d'assez bons résultats, ces méthodes le modifient très peu, juste pour créer un réseau fermé, et les résultats sont satisfaisants.

En ce qui concerne la méthode 3, qui crée un nouveau réseau de courbes en contraignant les longueurs, elle satisfait alors très bien les contraintes de longueur. Cela fournit également des courbes plus lisses, car ce ne sont pas des courbes modulées par des fonctions de Hermite. Elle donne des résultats d'erreur du même ordre que les précédentes méthodes, avec selon les cas, un avantage par rapport aux autres ou un léger désavantage. Cependant, cette méthode donne des courbes qui satisfont vraiment les contraintes de tangences et longueur pour tout le réseau : elle nous sera utile pour la suite, lorsque nous chercherons

à suivre une surface en déformation.

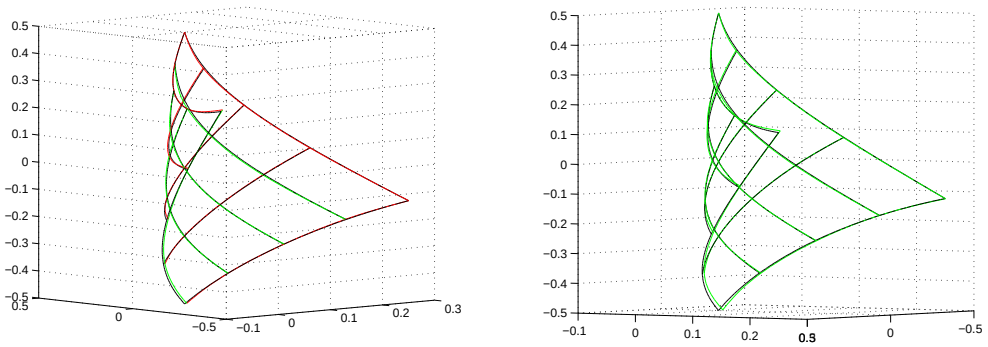
B. RÉSEAU DE COURBES POUR LES CONTRAINTES DE LONGUEUR DANS UNE SEULE DIRECTION

Voyons à présent le réseau de courbes que nous obtenons si nous utilisons les données acquises par un ensemble de rubans instrumentés dans la même direction. Nous rappelons que dans ce cas, nous obtenons les courbes dans la direction contrainte avec la méthode qui les crée indépendamment les unes des autres, puis nous créons les courbes dans l'autre sens avec les contraintes de tangentes (données par les capteurs) et les contraintes de positions (données par les courbes précédemment reconstruites). Nous allons les comparer avec une des méthodes du filet de capteurs (les données étant des données simulées, nous pouvons avoir les données de longueur que nous n'aurions pas avec un tel système et que nous ignorons pour la reconstruction).

Dans les exemples qui suivent, nous allons calculer diverses erreurs en séparant les résultats des deux directions. Les erreurs sont les erreurs de longueur et les erreurs de position globale des morceaux de courbes. Nous allons comparer les résultats obtenus par cette méthode avec la méthode 2 précédemment étudiée lorsque nous supposions avoir un filet de données.

Nous reprenons les mêmes surfaces tests que précédemment.

Portion de cône Le réseau de courbes obtenu est visible sur la figure FIG.3.21, où nous représentons à gauche les courbes avec contraintes de longueur en rouge et celles non contraintes en vert. Nous y superposons le réseau réel en noir, et nous la comparons au réseau reconstruit avec la méthode 2 à droite.



(a) Nouveau réseau reconstruit (b) Comparaison avec la méthode 2

FIG. 3.21 – Portion de cône : Reconstruction du réseau avec contraintes dans une direction, *(noir) : réseau réel, (gauche, rouge) : courbes reconstruites avec contraintes de longueur, (gauche vert) : courbes reconstruites avec contraintes de positions*

Nous obtenons visuellement une très bonne reconstruction du réseau de courbes. Voici les erreurs calculées pour chaque morceau de courbes (voir FIG.3.22) : erreurs de longueurs pour les morceaux des courbes contraintes (tous les morceaux de la première direction ($(5-1).5$) plus la première courbe de la deuxième direction (4 morceaux) : soit 24 au

total) ainsi que les erreurs dans la direction non contrainte (les $(5-1).(5-1) = 16$ restantes). Nous séparons également les erreurs de position globale selon les contraintes de reconstruction.

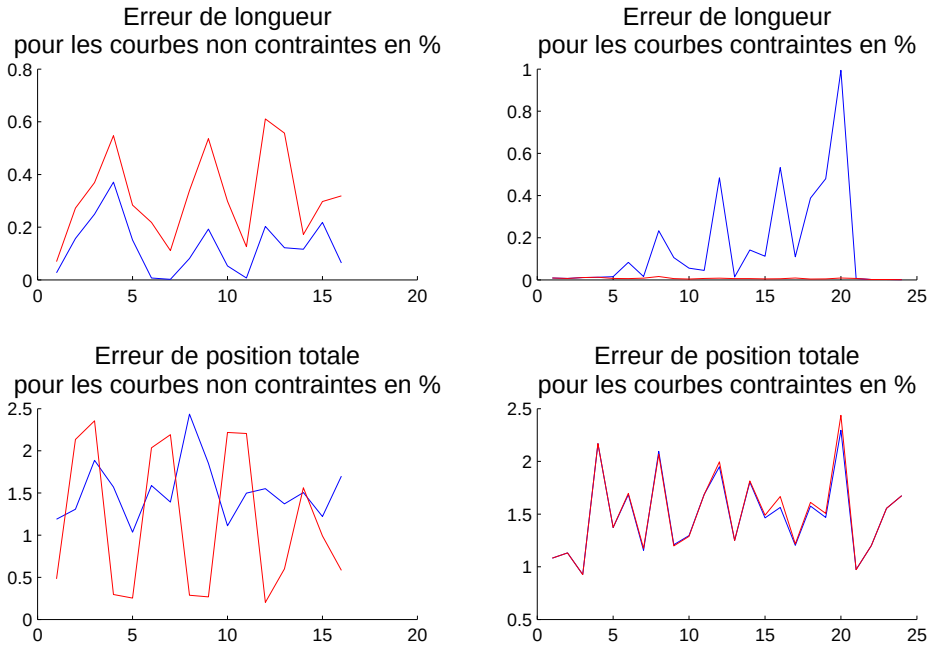
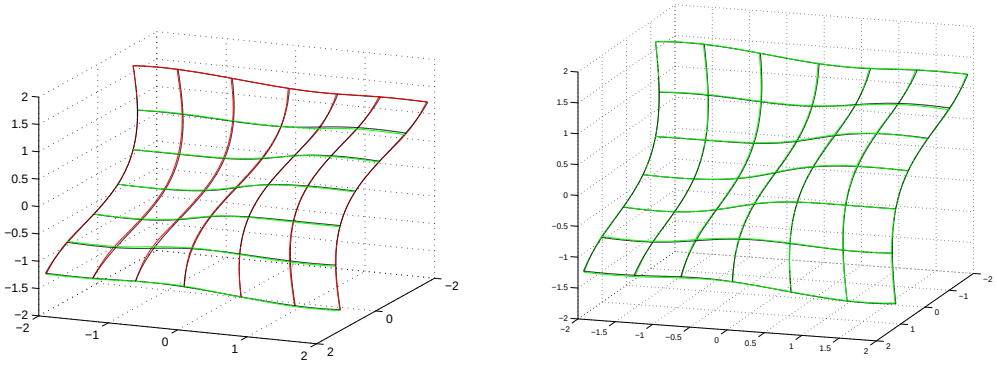


FIG. 3.22 – Portion de cône : Comparaison des systèmes d'acquisition, *(rouge) : erreurs de la reconstruction par rubans instrumentés, (bleu) : erreurs de la reconstruction par filet de capteurs*

D'après ces résultats, nous voyons clairement que les reconstructions des courbes avec les contraintes de longueur sont bien reconstruites (la méthode étant la méthode initiale). Il nous faut ici regarder les reconstructions dans l'autre sens, qui se basent seulement sur les positions des points des courbes de l'autre sens et des données tangentielles des points de mesure : nous n'avons aucune connnaissance sur les longueurs. Pourtant les erreurs de longueur sont très faibles pour cet exemple (0.6% maximum) et les erreurs de positions globales de la courbe oscillent autour des valeurs obtenues avec un filet de données, mais restent faibles.

Voyons les résultats obtenus sur les autres tests.

Vagues Voici le réseau de courbes obtenu pour le test 2 (FIG.3.23).



(a) Nouveau réseau reconstruit

(b) Comparaison avec la méthode 2

FIG. 3.23 – Vagues : Reconstruction du réseau avec contraintes dans une direction*(noir) : réseau réel, (gauche, rouge) : courbes reconstruites avec contraintes de longueur, (gauche vert) : courbes reconstruites avec contraintes de positions*

Cet exemple montre également un très bonne reconstruction du réseau. Voici les erreurs calculées pour chaque morceau de courbe (voir FIG.3.24) : erreurs pour les morceaux des courbes contraintes (soit 48 au total) ainsi que les erreurs dans la direction non contrainte (les 36 restantes).

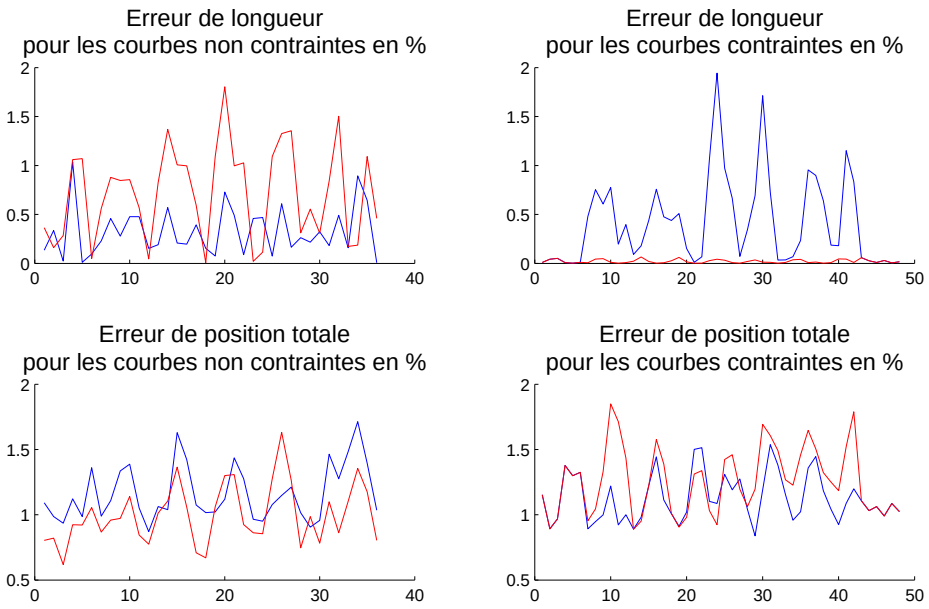
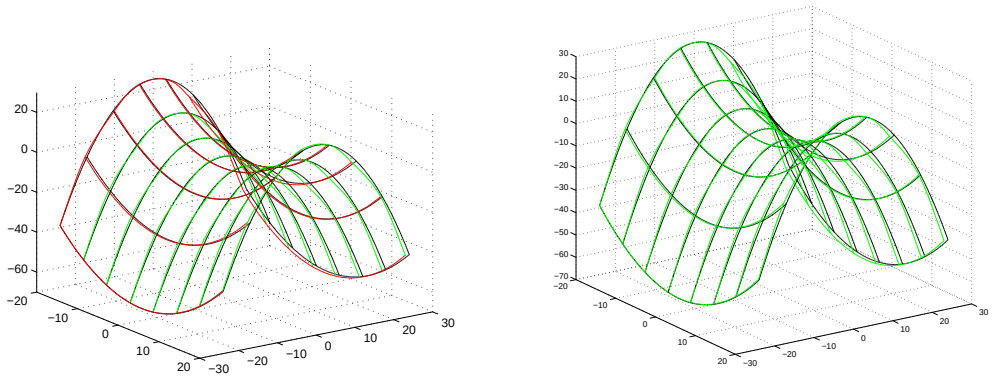


FIG. 3.24 – Vagues : Comparaison des systèmes d'acquisition, *(rouge) : erreurs de la reconstruction par rubans instrumentés, (bleu) : erreurs de la reconstruction par filet de capteurs*

Pour cet exemple, nous voyons que les résultats sont comparables pour les deux types d'acquisition de données : les erreurs de longueur sont meilleures dans une direction que dans l'autre (et vice versa pour l'autre méthode) et les résultats des erreurs globales de position sont similaires.

Selle Pour le test 3, voici le réseau de courbes obtenu sur la figure FIG.3.25.

Voici les erreurs calculées pour chaque morceau de courbe (voir FIG.3.26) : erreurs pour les morceaux des courbes contraintes (soit 63 morceaux) et les erreurs dans la direction non contrainte (49 morceaux).



(a) Nouveau réseau reconstruit (b) Comparaison avec la méthode 2

FIG. 3.25 – Selle : Reconstruction du réseau avec contraintes dans une direction*(noir) : réseau réel, (gauche, rouge) : courbes reconstruites avec contraintes de longueur, (gauche vert) : courbes reconstruites avec contraintes de positions*

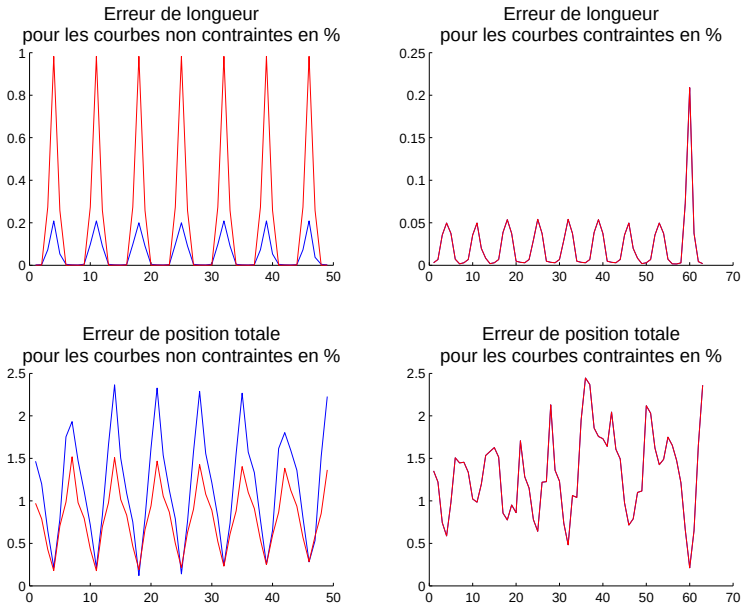
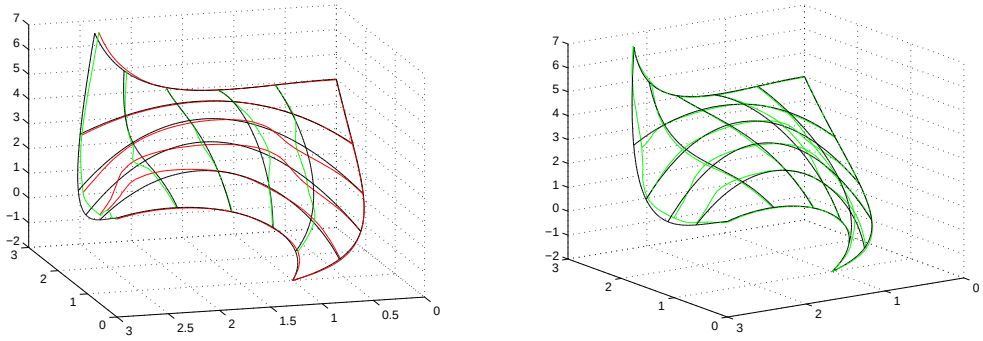


FIG. 3.26 – Selle : Comparaison des systèmes d'acquisition, *(rouge) : erreurs de la reconstruction par rubans instrumentés, (bleu) : erreurs de la reconstruction par filet de capteurs*

Dans cet exemple, nous voyons que les courbes dans la direction contrainte sont identiques (c'est ce que nous avions déjà vu lors des comparaisons des réseaux dans les tests précédents : en effet les courbes de longueur contrainte de la nouvelle méthode sont celles du réseau initial lorsque nous avons un filet de données). En ce qui concerne la direction non contrainte, nous avons des erreurs un peu plus élévees pour les erreurs de longueur (de

l'ordre de 1% maximum alors que l'autre est de 0.2%) mais les résultats de positions de courbes sont similaires voire meilleures pour cette méthode.

Surface de Bézier Le réseau de courbes obtenu pour le test 4 est représenté sur le graphique de la figure FIG.3.27.



(a) Nouveau réseau reconstruit (b) Comparaison avec la méthode 2

FIG. 3.27 – Surface de Bézier : Reconstruction du réseau avec contraintes dans une direction *(noir) : réseau réel, (gauche, rouge) : courbes reconstruites avec contraintes de longueur, (gauche vert) : courbes reconstruites avec contraintes de positions*

Voici les erreurs calculées pour chaque morceau de courbe (voir FIG.3.28) : erreurs pour les morceaux des courbes contraintes (35 au total) ainsi que les erreurs dans la direction non contrainte (les 25 morceaux restants).

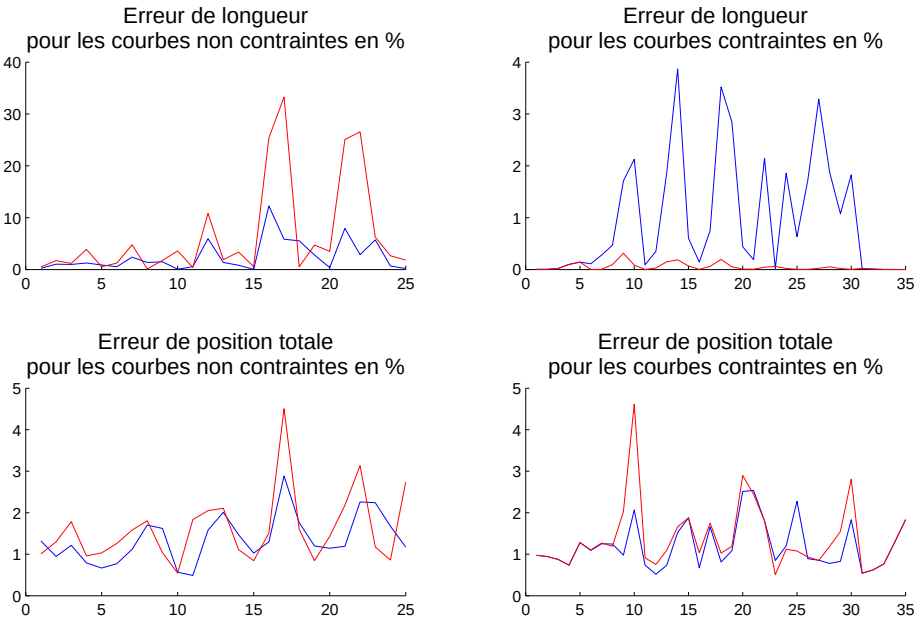


FIG. 3.28 – Surface de Bézier : Comparaison des systèmes d'acquisition, *(rouge) : erreurs de la reconstruction par rubans instrumentés, (bleu) : erreurs de la reconstruction par filet de capteurs*

Dans cet exemple, les erreurs de longueur sont assez importantes, mais au final les erreurs concernant les positions des courbes sont acceptables et de l'ordre des erreurs dans l'autre direction.

Commentaires Cette méthode de reconstruction lorsque nous avons pour données des courbes dans une seule direction donne de bons résultats. Les erreurs dans la direction non contrainte sont fortement liées aux erreurs dans la direction contrainte. En effet, nous avons vu que le fait d'avoir de bonnes reconstructions dans la direction contrainte donne alors de bonnes reconstructions des courbes dans l'autre direction (car dans ce cas, les positions des capteurs sont proches des positions réelles). Donc pour avoir une bonne reconstruction de ce genre de réseau, nous devons avoir au départ un bonne répartition des capteurs sur la surface. Dans ce cas, la création des courbes dans la direction non contrainte (qui dépend des courbes de la première direction et qui minimise leur énergie) donne des résultats comparables à la donnée d'un filet de données.

Nous allons à présent nous intéresser aux surfaces reconstruites à partir de ces différents réseaux de courbes, et avancer ainsi dans la comparaison des diverses méthodes d'acquisition et de reconstruction.

3.6.2 RECONSTRUCTION DES SURFACES

A. ERREURS CALCULÉES

Nous allons ici calculer des erreurs caractéristiques des surfaces qui sont :

– erreur sur la position des capteurs :

$$E_{capt} = \left(\frac{1}{L_{ref}.\,(n_u+1)\,(n_v+1)} \left(\sum_{k_u=0}^{n_u} \sum_{k_v=0}^{n_v} \left\| S_{th}\left(\frac{k_u}{n_u}, \frac{k_v}{n_v}\right) - S_{obt}\left(\frac{k_u}{n_u}, \frac{k_v}{n_v}\right) \right\| \right) \right) . 100\,\%$$

avec L_{ref} longueur de référence de la surface (la plus grande des deux dimensions)

– erreur de Hausdorff modifiée (pour un sur-échantillonnage de la surface en $(N_u + 1).(N_v + 1)$ points) :

$$E_{Haus} = \frac{1}{L_{ref}} \max \left\{ \begin{array}{c} \max_{k_u,k_v} \left(\min_{j_u,j_v} \left\| S_{th}\left(\frac{k_u}{N_u}, \frac{k_v}{N_v}\right) - S_{obt}\left(\frac{j_u}{N_u}, \frac{j_v}{N_v}\right) \right\| \right), \\ \max_{k_u,k_v} \left(\min_{j_u,j_v} \left\| S_{th}\left(\frac{j_u}{N_u}, \frac{j_v}{N_v}\right) - S_{obt}\left(\frac{k_u}{N_u}, \frac{k_v}{N_v}\right) \right\| \right) \end{array} \right\} . 100\,\%$$

B. QUELQUES RÉSULTATS

Nous allons reprendre les diverses surfaces tests que nous avons étudié jusqu'à présent. Pour chacune d'entre elles, nous allons reprendre les quatre méthodes de reconstruction (les trois méthodes développées si nous avons pour données un filet de capteurs, et la dernière méthode qui reconstruit le réseau de courbes à partir de données filaires dans une seule direction).

Portion de cône Reprenons le morceau de cône instrumenté par une grille de cinq capteurs par cinq (voir FIG.3.29).

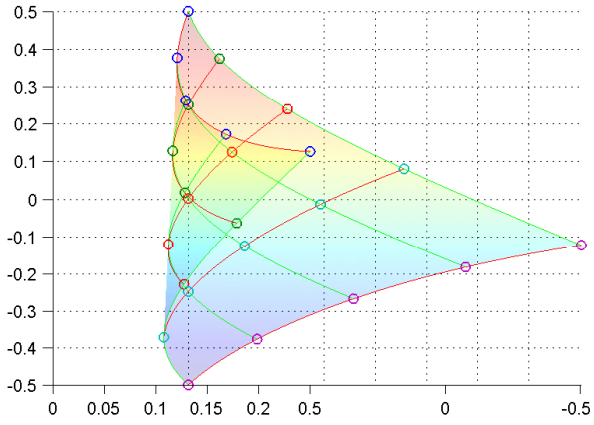
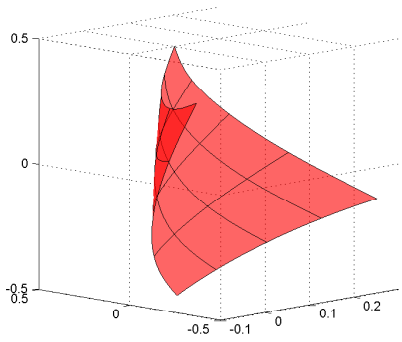


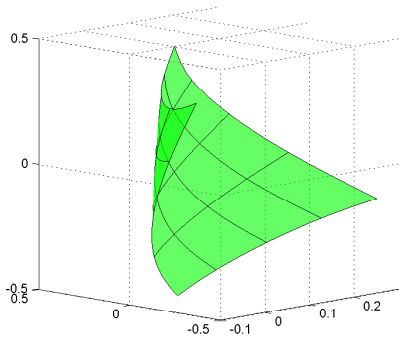
FIG. 3.29 – Portion de cône : réseau à reconstruire

Voyons à présent les quatre surfaces reconstruites à partir des données des capteurs. Sur la figure FIG.3.30, nous voyons en 3.30(a) la surface reconstruite à partir de la méthode 1 de

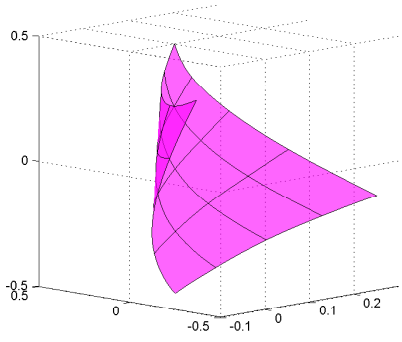
reconstruction (qui ferme le réseau en modifiant dès le départ tous les points d'intersection), sur 3.30(b) celle obtenue par la méthode 2 (qui modifie progressivement les positions des capteurs), sur 3.30(c) nous voyons la surface obtenue avec le réseau de courbes qui satisfait les contraintes de longueur dans les deux directions, et enfin dans 3.30(d) nous voyons la surface obtenue à partir des rubans instrumentés dans une seule direction. Pour chaque surface, nous voyons superposé son réseau de courbes.



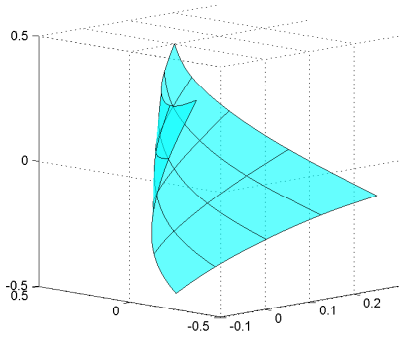
(a) Filet : méthode 1



(b) Filet : méthode 2



(c) Filet : méthode 3



(d) Données dans 1 direction

FIG. 3.30 – Portion de cône : Reconstructions de la surface selon les méthodes

Nous obtenons visuellement des surfaces très semblables entre elles et ressemblant à la surface à reconstruire. Voyons les erreurs que nous avons calculées sur ces surfaces : l'erreur de position de capteurs ainsi que l'erreur de Hausdorff sont rassemblées dans le tableau suivant (voir TAB.3.1).

Méthode utilisée	E_{capt}	E_{Haus}
Filet Méthode 1	0.31%	2.0%
Filet Méthode 2	0.32%	2.01%
Filet Méthode 3	0.38%	1.34%
Dans une seule direction	0.31%	1.40%

TAB. 3.1 – Erreurs de reconstruction associées à la portion de cône

Ces erreurs confirment les bons résultats visuels : les positions des capteurs reconstruits sont très proches des positions réelles (autour de 0.3%), et les erreurs de Hausdorff sont également faibles (les points les plus éloignés entre la surface réelle et les surfaces simulées le sont d'une distance inférieure à 2% (par rapport à une longueur de référence de la surface)). La solution la meilleure ici lorsque nous avons un filet de données est celle qui a pour contraintes les contraintes de longueur (c'est déjà ce que nous observions lors de la comparaison des réseaux de courbes), et elle est comparable à la méthode qui ne dispose que de données de longueur dans une seule direction.

Voyons à présent ce que nous obtenons pour les autres surfaces à reconstruire.

Vagues Reprenons à présent la surface vagues (voir FIG.3.31).

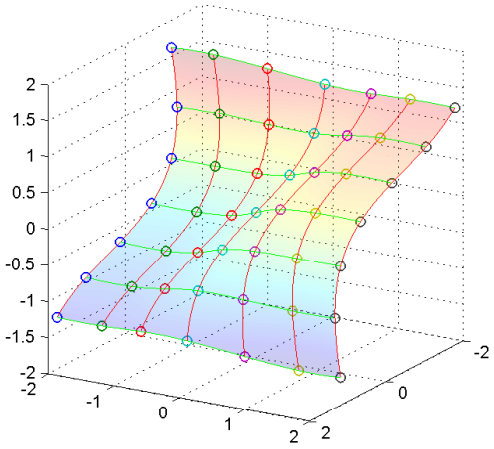
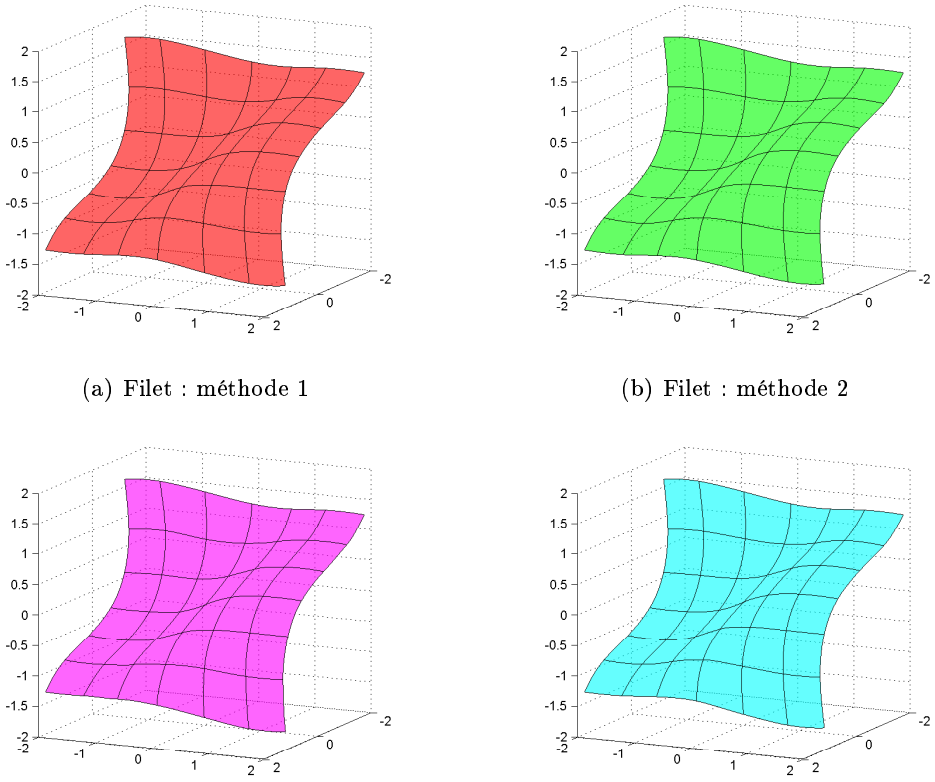


FIG. 3.31 – Vagues : réseau à reconstruire

Les quatre surfaces reconstruites à partir des données des capteurs sont sur la figure FIG.3.32.



(a) Filet : méthode 1

(b) Filet : méthode 2

(c) Filet : méthode 3

(d) Données dans 1 direction

FIG. 3.32 – Vagues : Reconstructions de la surface selon les méthodes

Nous obtenons également de bonnes reconstructions sur cet exemple. Les erreurs de positions de capteurs ainsi que l'erreur de Hausdorff sont dans le tableau suivant (voir TAB.3.2).

Méthode utilisée	E_{capt}	E_{Haus}
Filet Méthode 1	0.35%	1.52%
Filet Méthode 2	0.36%	1.61%
Filet Méthode 3	0.78%	1.82%
Dans une seule direction	0.45%	1.52%

TAB. 3.2 – Erreurs de reconstruction associées aux vagues

Les erreurs sont toujours aussi faibles : toutes les méthodes sont satisfaisantes. Nous notons tout de même un léger désavantage pour la méthode qui force les contraintes de longueur (méthode 3), c'est également ce que nous observions sur les réseaux de courbes. Nous notons aussi le fait que la méthode qui n'utilise que les données de longueur dans une seule direction nous permet d'obtenir d'aussi bons résultats que les autres méthodes.

Selle Le test suivant est le paraboloïde hyperbolique avec un réseau de capteurs de taille huit par huit (voir FIG.3.33).

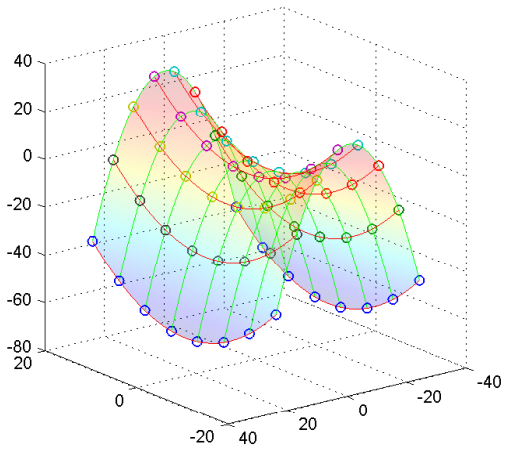
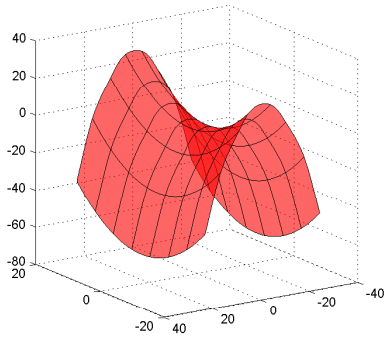
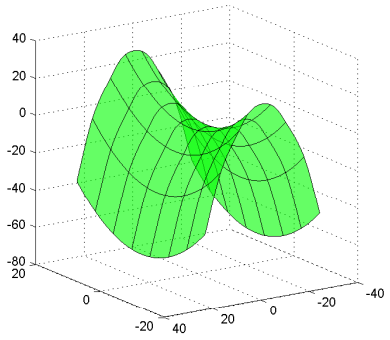


FIG. 3.33 – Selle : réseau à reconstruire

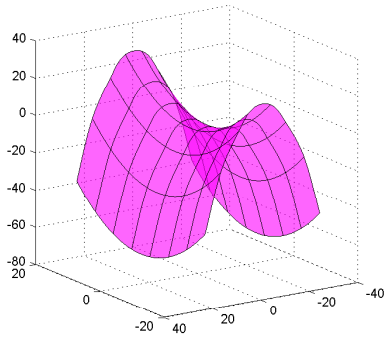
Les résultats des surfaces obtenues sont dans la figure FIG.3.34, où nous voyons les surfaces et leur réseau superposé.



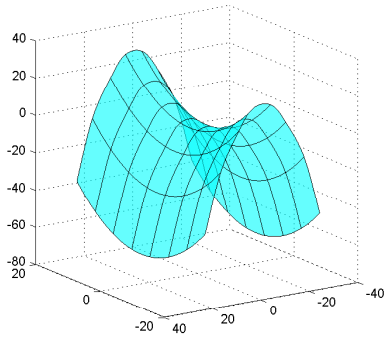
(a) Filet : méthode 1



(b) Filet : méthode 2



(c) Filet : méthode 3



(d) Données dans 1 direction

FIG. 3.34 – Selle 3 : Reconstructions de la surface selon les méthodes

Les erreurs de position des capteurs ainsi que l'erreur de Hausdorff sont rassemblées dans le tableau suivant (voir TAB.3.3)

Méthode utilisée	E_{capt}	E_{Haus}
Filet Méthode 1	0.35%	1.57%
Filet Méthode 2	0.35%	1.57%
Filet Méthode 3	0.35%	1.0%
Dans une seule direction	0.35%	1.03%

TAB. 3.3 – Erreurs de reconstruction associées à la selle

En ce qui concerne cet exemple, nous avions vu lors de la reconstruction des réseaux que nous obtenions des erreurs presque identiques en ce qui concernait la position des capteurs : c'est bien sûr toujours ce que nous observons ici. Nous avions également noté de plus faibles erreurs de reconstruction globale des morceaux de courbes avec la méthode 3 : ceci se confirme ici avec une erreur de Hausdorff meilleure avec cette méthode-ci que les deux autres. La méthode développée lorsque nous n'avons qu'une seule famille de courbes dans la même direction donne des erreurs du même ordre que la meilleure méthode dans l'autre cas.

Surface de Bézier Le dernier exemple montre la surface de Bézier avec son réseau de capteurs de taille six par six. (voir FIG.3.35)

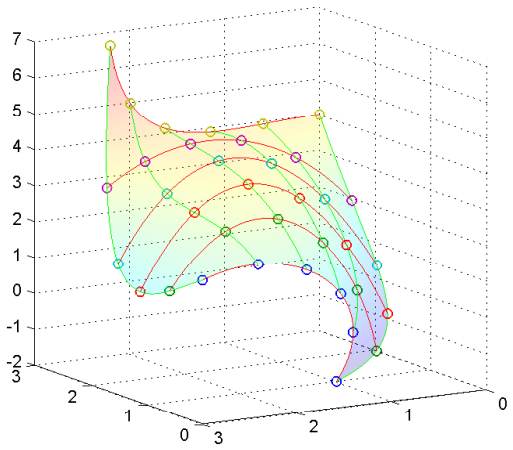
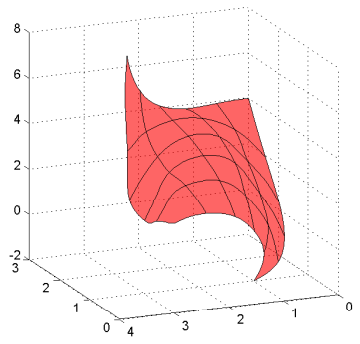
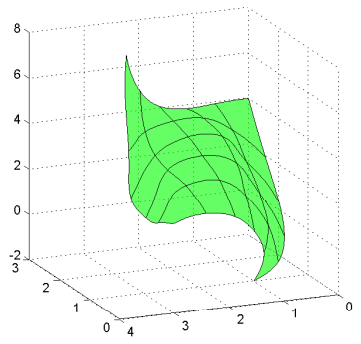


FIG. 3.35 – Surface de Bézier : réseau à reconstruire

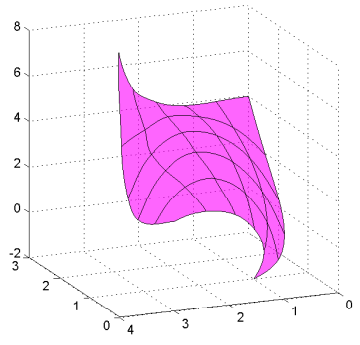
Nous obtenons les quatre surfaces de la figure FIG.3.36 selon les méthodes employées.



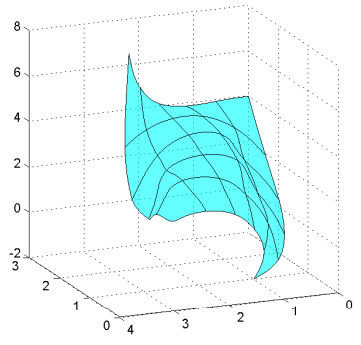
(a) Filet : méthode 1



(b) Filet : méthode 2



(c) Filet : méthode 3



(d) Données dans 1 direction

FIG. 3.36 – Surface de Bézier : Reconstructions de la surface selon les méthodes

Cet exemple-ci montrait des réseaux de courbes assez perturbés, cela étant dû au fait que le réseau initial était assez éloigné des courbes réelles. Nous voyons maintenant les conséquences que cela a sur les surfaces reconstruites à partir de ces réseaux. Nous voyons que les surfaces issues des méthodes 1 et 2 ne semblent pas lisses, contrairement à la surface issue de la méthode 3 (c'est aussi ce que nous observions sur les courbes elles-mêmes). Voyons les erreurs engendrées par ces surfaces dans le tableau TAB.3.4.

Méthode utilisée	E_{capt}	E_{Haus}
Filet Méthode 1	0.56%	2.53%
Filet Méthode 2	0.72%	2.52%
Filet Méthode 3	0.77%	2.54%
Dans une seule direction	0.88%	4.31%

TAB. 3.4 – Erreurs de reconstruction associées à la surface de Bézier

Nous observons de bonnes positions de capteurs (inférieures au 1% d'erreur). En ce qui concerne les erreurs de Hausdorff, les trois méthodes qui utilisent un filet de données donnent des erreurs similaires ; en ce qui concerne la méthode des courbes dans une seule direction, elle donne une erreur plus importante : cette façon d'acquérir des données pour

cet exemple n'est pas la plus appropriée.

Nous pouvons cependant noter que dans cet exemple, les courbes isoparamétriques à reconstruire sont de vraies courbes gauches, ce qui n'est pas forcément le cas des quelques exemples précédents où la torsion semble assez faible. Cela nécessiterait alors plus de capteurs pour une meilleure reconstruction.

Commentaires Ces exemples montrent que toutes les méthodes que nous avons développées donnent de très bonnes reconstructions de surfaces à partir de ces capteurs répartis sur la surface. Les erreurs de positions des capteurs sont toujours faibles (cela est dû à la méthode initiale de reconstruction des courbes, car nos capteurs sont définis comme les intersections des courbes sur la surface). Les erreurs globales de positionnement de la surface sont également faibles, inférieures à 2% si nous avons une bonne répartition des capteurs sur la surface (en effet, le dernier test montre que si nous n'avons pas assez de capteurs, les reconstructions aboutissent à des surfaces qui ne sont pas proches des surfaces à reconstruire).

3.6.3 RECONSTRUCTION DES SURFACES EN FONCTION DU NOMBRE DE CAPTEURS

Nous allons nous intéresser ici à l'évolution des erreurs lorsque nous augmentons la densité des capteurs sur la surface à reconstruire. Nous allons donc reprendre les divers exemples vus jusqu'à présent.

Nous montrerons ici en plus de l'erreur de position des capteurs et l'erreur de Hausfdorff, l'erreur moyenne des longueurs des courbes du réseau :

$$E_{lon} = \frac{1}{(n_u+1)+(n_v+1)} \left(\sum_{k_u=0}^{n_u} \frac{\left| L^u_{k_u,th} - L^u_{k_u,obt} \right|}{L^u_{k_u,th}} + \sum_{k_u=0}^{n_u} \frac{\left| L^v_{k_v,th} - L^v_{k_u,obt} \right|}{L^v_{k_v,th}} \right) . \, 100 \, \%$$

avec $L^u_{k_u,th}$ longueur de la courbe $C^u_{k_u,th}$ courbe isoparamétrique en u numero k_u de la surface théorique.

PORTION DE CÔNE

Nous reprenons ici le premier test qui est le morceau de cône. Nous allons faire varier le nombre de capteurs dans les deux directions avec les valeurs suivantes : 3, 4, 5, 7, 9, 11, 14 (nous aurons alors des réseaux de capteurs de taille 3 par 3, à 14 par 14). Nous appliquons les quatre méthodes que nous avons développé, et nous calculons les trois erreurs que nous avons explicité précédemment. Nous voyons sur la figure FIG.3.37 les courbes d'erreurs associées aux diverses méthodes : en rouge la méthode 1 (qui fixe tous les points d'intersection dès le départ), en vert la méthode 2 (qui fixe les points de croisement progressivement), en rose la méthode 3 (qui cherche les points d'intersection en contraignant les longueurs), et en bleu les erreurs issues de la méthode 4, qui prend en compte les contraintes de longueur dans une seule direction.

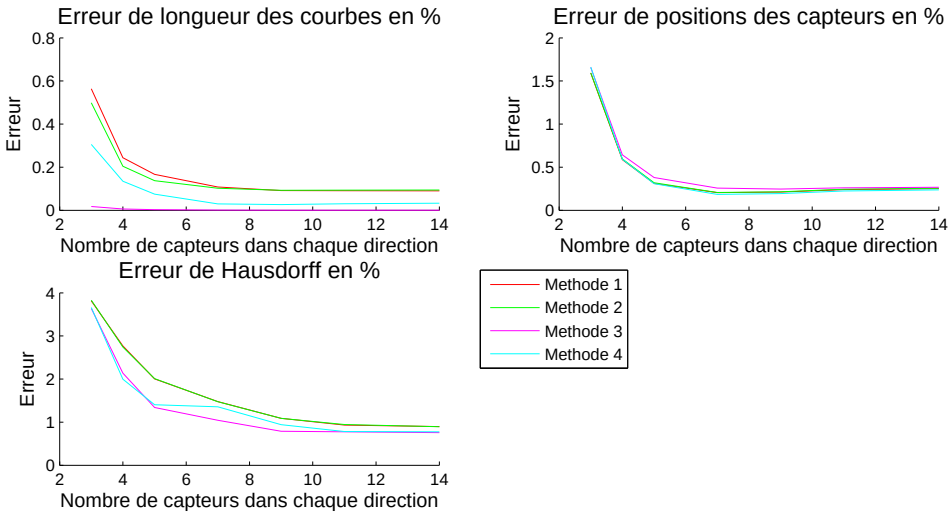


FIG. 3.37 – Portion de cône : Erreurs en fonction du nombre de capteurs

VAGUES

Pour le deuxième test, nous appliquons nos diverses méthodes pour les nombres de capteurs suivants : 4, 5, 6, 8, 10, 13 et 16. Nous pouvons comparer les erreurs sur la figure FIG.3.38.

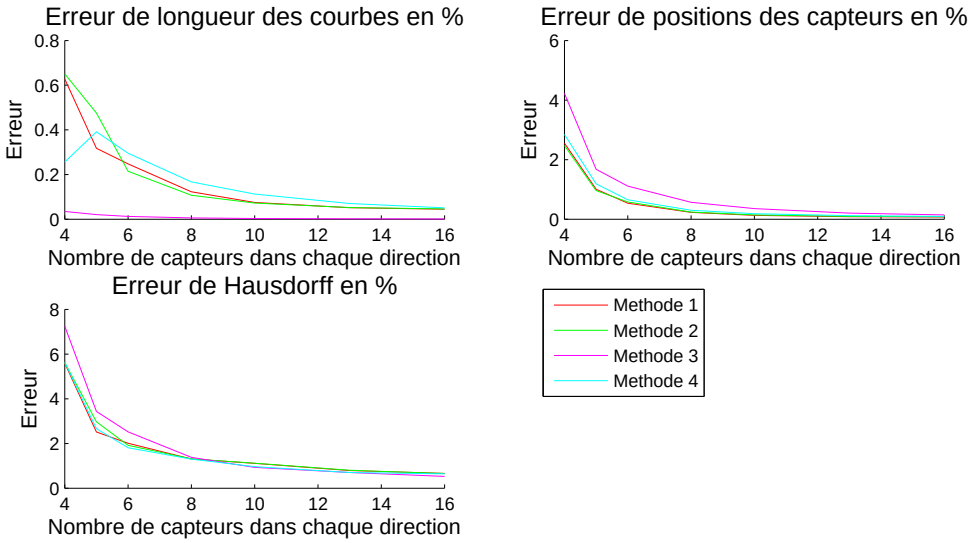


FIG. 3.38 – Vagues : Erreurs en fonction du nombre de capteurs

SELLE

Nous reconstruisons la surface-test 3 pour une grille de capteurs dont la taille varie avec les valeurs suivantes : 7, 8, 9, 11, 13, 16 et 19. Les erreurs sont représentées sur la figure FIG.3.39. Dans cet exemple, les erreurs des méthodes 1 et 2 sont identiques : leurs

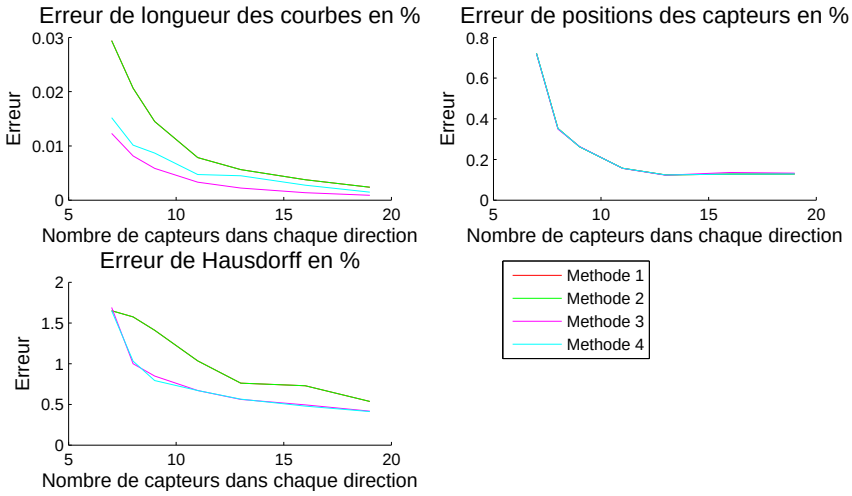


FIG. 3.39 – Selle : Erreurs en fonction du nombre de capteurs

courbes d'erreurs sont confondues (rouge et vert). Les erreurs de positions des capteurs sont également confondues pour les quatre méthodes mises en oeuvre ici.

SURFACE DE BÉZIER

Pour le dernier exemple, nous appliquons le nombre suivant de capteurs dans chaque direction : 4, 5, 6, 8, 10, 13 et 16. Voyons les erreurs obtenues sur la figure FIG.3.40.

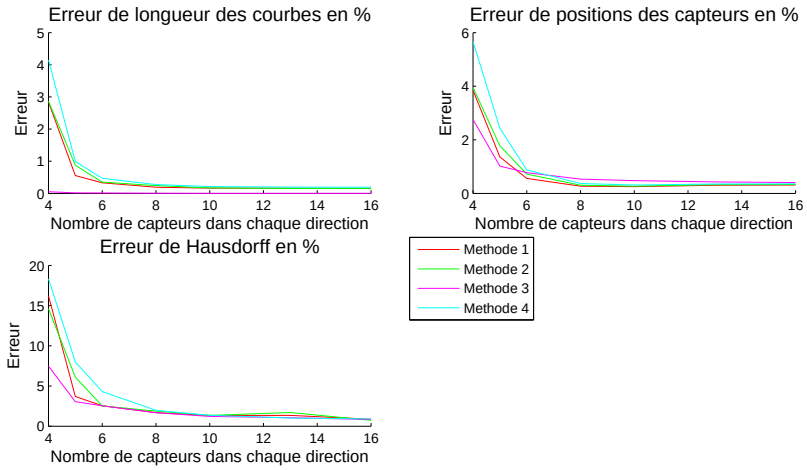


FIG. 3.40 – Surface de Bézier : Erreurs en fonction du nombre de capteurs

COMMENTAIRES

Sur toutes ces courbes d'erreurs, nous voyons une forte décroissance puis une stagnation, et cela pour toutes les méthodes développées. Ces courbes montrent que nous pourrons obtenir de très bonnes reconstructions de surfaces (avec des erreurs très satisfaisantes) pour un nombre assez peu élevé de capteurs.

3.7 CONCLUSION

Dans ce chapitre, nous avons répondu à la question des reconstructions de surfaces à partir de données tangentielles. La première étape a consisté à définir le problème, nous avons vu la nécessité de nous appuyer sur un réseau de courbes tracées sur la surface, et qui prolonge ainsi le travail effectué dans la première partie de ce mémoire. Nous avons décelé deux méthodes différentes d'acquisition : une acquisition par un filet de capteurs, qui décrit ainsi deux familles de courbes, s'intersectant aux positions des capteurs, donnant les orientations des tangentes le long de ces deux courbes, ainsi que les répartitions de ces noeuds capteurs ; et une deuxième méthode d'acquisition étant constitué par une seule famille de courbes dans une direction, donnant également les données tangentielles et la répartition de ces capteurs le long des courbes mais aussi les données des vecteurs normaux à ces courbes sur la surface.

Cette définition du problème nous a permis de mettre en oeuvre la stratégie suivante : tout d'abord reconstruire un réseau de courbes isoparamétriques dans les deux directions sur la surface (cette étape étant dépendante de la méthode d'acquisition utilisée pour obtenir les données), puis définir la surface sur laquelle s'appuie ce réseau de courbes par une méthode de remplissage des patchs ainsi définis. Nous avons comparé les différentes méthodes que nous avons mises au point, tant au niveau des courbes obtenues, que des surfaces reconstruites, et nous avons pu en conclure que ces méthodes nous permettaient d'obtenir de bonnes reconstructions de surfaces. Nous avons également étudié sur des exemples l'évolution des erreurs de reconstruction en fonction du nombre de capteurs, pour nous rendre compte du caractère convergent de ces méthodes de reconstruction.

Le problème qui se pose à nous à présent est la reconstruction de la déformation d'une surface au cours du temps : c'est le sujet que nous traitons dans le chapitre suivant.

Chapitre 4

Déformation de surfaces

Ce chapitre traite la question de la reconstruction au cours du temps d'une surface en déformation. Nous devons ainsi retrouver à chaque pas de temps la forme d'une surface satisfaisant les données des capteurs. Une possibilité serait de réutiliser la méthode vue dans le chapitre précédent, mais cela n'est pas une méthode optimale. En effet, si nous résumons la méthodologie de résolution du problème pour un filet de capteurs, nous devons définir un premier réseau, le modifier pour recaler les courbes et construire la surface. Etant donné que nous connaissons la position de la surface à l'instant précédent, nous avons déjà un réseau initial que nous pouvons déformer pour qu'il satisfasse les nouvelles contraintes fournies pas les capteurs.
Nous allons ainsi définir une nouvelle méthode de suivi de déformation de surfaces dans le temps, qui consiste dans un premier temps à déformer le réseau de courbes connu à l'instant précédent (nous avons déjà beaucoup d'outils à notre disposition, vus notamment lors du suivi de déformation de courbes, où lors des recalages de réseaux pour la reconstruction de surfaces), puis nous définirons la surface qui s'appuie sur le nouveau réseau de courbes calculé (avec la même méthode que dans le chapitre précédent).
Nous illustrerons les méthodes par deux exemples de surfaces en mouvement, que nous caractériserons par l'étude des erreurs obtenues.

Sommaire

Nous allons ici nous intéresser au suivi de déformation d'une surface, tout comme nous l'avons étudié pour le problème de reconstruction des courbes. Dans le cas de reconstruction d'une surface, nous supposons qu'en plus de la connaissance des données acquises par un système de courbes sur la surface (données tangentielles et répartition des capteurs), nous connaissons la position de la surface à l'instant précédent.

La surface étant déterminée par son réseau de courbes, nous allons procéder de la manière suivante : nous allons déformer le réseau de courbes en fonction de la position à l'instant précédent et des nouvelles données tangentielles, puis nous créerons la nouvelle surface avec la méthode de Coons, comme nous l'avons fait dans le chapitre précédent. Voyons comment déformer le réseau de courbes existant pour qu'il satisfasse les nouvelles données tangentielles.

4.1 Déformation du réseau de courbes

Nous cherchons un nouveau réseau de courbes

$$\left\{\widetilde{C^u_{k_u}}(v), k_u = 0, ..., n_u\right\} \text{ et } \left\{\widetilde{C^v_{k_v}}(u), k_v = 0, ..., n_v\right\}$$

proches de l'ancien réseau

$$\left\{C^u_{k_u}(v), k_u = 0, ..., n_u\right\} \text{ et } \left\{C^v_{k_v}(u), k_v = 0, ..., n_v\right\}$$

qui satisfait les nouvelles contraintes données par les capteurs. Les données acquises dépendent du système de mesure que nous utilisons (filet de données ou rubans instrumentés dans une seule direction), nous allons donc résoudre indépendamment ces deux problèmes.

4.1.1 Filet de données

• Si le système d'acquisition est un filet de capteurs, l'ancien réseau satisfaisait les conditions suivantes :

- les courbes $\left\{C^u_{k_u}(v), k_u = 0, ..., n_u\right\}$ sont tangentes aux vecteurs

$$T^u(k_u, k_v)k_v = 0, ..., n_v$$

  aux paramètres $v = \frac{k_v}{n_v}$,
- les distances curvilignes le long des courbes $\left\{C^u_{k_u}(v), k_u = 0, ..., n_u\right\}$ entre capteurs adjacents (k_u, k_v) et $(k_u, k_v + 1)$ sont de $L^u_{k_u}(k_v), k_v = 0, ..., n_v - 1$,
- les courbes $\left\{C^v_{k_v}(u), k_v = 0, ..., n_v\right\}$ sont tangentes aux vecteurs

$$T^v(k_u, k_v)k_u = 0, ..., n_u$$

  aux paramètres $u = \frac{k_u}{n_u}$,

– les distances curvilignes le long des courbes $\left\{C^v_{k_v}(u), k_v = 0, ..., n_v\right\}$ entre capteurs adjacents (k_u, k_v) et $(k_u + 1, k_v)$ sont de $L^v_{k_v}(k_u), k_u = 0, ..., n_u - 1$.

• Nous cherchons le nouveau réseau qui doit satisfaire les mêmes données de distances curvilignes entre les capteurs, mais doit répondre aux nouvelles données tangentielles :

– les courbes $\left\{\widetilde{C^u_{k_u}}(v), k_u = 0, ..., n_u\right\}$ doivent être tangentes aux vecteurs

$$\widetilde{T^u}(k_u, k_v)k_v = 0, ..., n_v$$

aux paramètres $v = \frac{k_v}{n_v}$,

– les courbes $\left\{\widetilde{C^v_{k_v}}(u), k_v = 0, ..., n_v\right\}$ doivent être tangentes aux vecteurs

$$\widetilde{T^v}(k_u, k_v)k_u = 0, ..., n_u$$

aux paramètres $u = \frac{k_u}{n_u}$.

Une solution serait de résoudre indépendamment ces problèmes pour chaque courbe comme nous l'avons résolu dans le chapitre de déformation de courbes. Cependant, nous n'obtiendrions pas de réseau fermé, ce dont nous avons besoin pour créer la surface qui s'appuie sur toutes ces courbes. Il nous faudrait alors utiliser une des méthodes de fermeture de réseau dont nous avons parlé dans le chapitre précédent, ce qui ne serait pas une façon optimale de résoudre le problème.

Nous allons opter ici pour la création d'un réseau fermé dès le départ. Nous allons ainsi chercher un nouveau réseau de courbes proche de l'ancien, qui satisfait les contraintes de longueur et de tangentes, et forme un ensemble de courbes fermé. Voyons ceci en détail dans la suite de ce document.

A. Création des deux premières courbes de bord

Nous allons en premier lieu définir les deux courbes de bord $\left(\widetilde{C^u_0}(v) \text{ et } \widetilde{C^v_0}(u)\right)$. Elles ont comme contrainte les données tangentielles et des données de distances curvilignes inter-capteurs.

Ces deux courbes sont construites avec des contraintes propres et indépendantes des autres données. Nous les reconstruisons alors indépendamment des autres : nous utilisons la méthode de déformation de courbes vue lors du chapitre sur les déformations de courbes.

Nous allons créer la nouvelle courbe $\widetilde{C^u_0}(v)$ dans la base des fonctions de Hermite :

$$\begin{aligned} \widetilde{C^u_0}(v) &= \widetilde{P}_{k_v}\varphi_0(n_v.v - k_v) + \widetilde{P}_{k_v+1}\varphi_1(n_v.v - k_v) \\ &+ \frac{1}{n_v}.\widetilde{T^u}(0, k_v)\varphi_2(n_v.v - k_v) + \frac{1}{n_v}.\widetilde{T^u}(0, k_v + 1)\varphi_3(n_v.v - k_v) \end{aligned} \quad (4.1)$$

pour v tel que $\frac{k_v}{n_v} \leq v \leq \frac{k_v+1}{n_v}$. Les inconnues sont ici les $\left\{\widetilde{P}_{k_v}, k_v = 1, ..., n_v\right\}$ positions des points des capteurs. Nous connaissons cependant leur positions au temps précédent

$\left\{P_{k_v} = C_0^u\left(\frac{k_v}{n_v}\right)\right\}$.

Les inconnues seront déterminées par résolution des contraintes de longueur morceau par morceau et minimisation d'énergie (revoir la démarche de calcul de la section 2.1.3 page 90).

Nous obtenons par cette méthode la définition de la courbe $\widetilde{C_0^u}(v)$ morceau par morceau. Il en sera de même pour la courbe $\widetilde{C_0^v}(u)$.

Nous avons ainsi les points de départ de toutes les autres courbes du réseau, les points de départ de toutes les isoparamétriques en u étant :

$$\widetilde{C_{k_u}^u}(0) = \widetilde{C_0^v}\left(\frac{k_u}{n_u}\right), k_u = 1, ..., n_u$$

et les points de départ des isoparamétriques en v étant :

$$\widetilde{C_{k_v}^v}(0) = \widetilde{C_0^u}\left(\frac{k_v}{n_v}\right), k_v = 1, ..., n_v.$$

B. Création des autres courbes du réseau

Nous devons à présent reconstruire les autres courbes du réseau $\left\{\widetilde{C_{k_u}^u}(v), k_u = 1, ..., n_u\right\}$ et $\left\{\widetilde{C_{k_v}^v}(u), k_v = 1, ..., n_v\right\}$ de sorte qu'elles s'intersectent aux données capteurs :

$$\widetilde{C_{k_u}^u}\left(\frac{k_v}{n_v}\right) = \widetilde{C_{k_v}^v}\left(\frac{k_u}{n_u}\right), k_u = 1, ..., n_u, k_v = 1, ..., n_v,$$

et sachant qu'elles sont proches des courbes à l'instant précédent. C'est exactement le même problème que lors du recalage du réseau avec contraintes de longueur, c'est-à-dire que nous construisons les courbes morceau par morceau, en cherchant les points d'intersection du réseau sous la contrainte de conservation des longueurs inter-capteurs.

Voici l'algorithme utilisé :

pour k_u de 1 à n_u

  pour k_v de 1 à n_v

    définir les courbes $\widetilde{C_{k_u}^u}$ sur $\left[\frac{k_v-1}{n_v}, \frac{k_v}{n_v}\right]$ et $\widetilde{C_{k_v}^v}$ sur $\left[\frac{k_u-1}{n_u}, \frac{k_u}{n_u}\right]$ en déterminant leur nouveau point d'arrivée commun $\widetilde{P}_{k_u,k_v}$ tel que leurs tangentes aux extrémités, leur longueur et leur point de départ soient fixés.

Les deux morceaux de courbes sont définis dans la base de Hermite :

$$\begin{aligned}
\widetilde{C_{k_u}^u}\left(\frac{k_v-1}{n_v} + \frac{t}{n_v}\right) &= \widetilde{P}_{k_u,k_{v-1}}.\varphi_0(t) + \widetilde{P}_{k_u,k_v}.\varphi_1(t) \\
&+ \widetilde{T^u}(k_u, k_{v-1}).\varphi_2(t) + \widetilde{T^u}(k_u, k_v).\varphi_3(t),\ t \in [0,1] \qquad (4.2) \\
\widetilde{C_{k_v}^v}\left(\frac{k_u-1}{n_u} + \frac{t}{n_u}\right) &= \widetilde{P}_{k_{u-1},k_v}.\varphi_0(t) + \widetilde{P}_{k_u,k_v}.\varphi_1(t) \\
&+ \widetilde{T^v}(k_{u-1}, k_v).\varphi_2(t) + \widetilde{T^v}(k_u, k_v).\varphi_3(t),\ t \in [0,1] \qquad (4.3)
\end{aligned}$$

L'inconnue est le point d'arrivée commun, $\widetilde{P}_{k_u,k_v}$, qui sera déterminée par les contraintes de longueur des deux morceaux de courbes, et par une minimisation d'énergie (revoir les détails dans section 3.3.2 en page 122).

Nous obtenons ainsi un réseau de courbes recalé, satisfaisant les contraintes de longueurs inter-capteurs, et les données tangentielles aux points des capteurs. Nous pouvons définir la surface solution par la méthode de remplissage de Coons.

Nous allons à présent résoudre le problème dans le cas où le système d'acquisition que nous utilisons n'est plus un filet de capteurs, mais est seulement constitué de rubans instrumentés dans la même direction.

4.1.2 Rubans instrumentés

• Si le système d'acquisition est constitué de rubans instrumentés dans la même direction, l'ancien réseau satisfaisait les conditions suivantes :

- les courbes $\left\{C^u_{k_u}(v), k_u = 0, ..., n_u\right\}$ sont tangentes aux vecteurs
$$T^u(k_u, k_v)k_v = 0, ..., n_v$$
aux paramètres $v = \frac{k_v}{n_v}$,
- les distances curvilignes le long de la courbe $\{C^u_0(v)\}$ entre capteurs adjacents $(0, k_v)$ et $(0, k_v + 1)$ sont de $L^u_0(k_v), k_v = 0, ..., n_v - 1$ (nous ne connaissons la répartition des capteurs que pour la première courbe dans cette direction : c'est la courbe qui lie les points de départ de toutes les courbes de l'autre direction),
- les courbes $\left\{C^v_{k_v}(u), k_v = 0, ..., n_v\right\}$ sont tangentes aux vecteurs
$$T^v(k_u, k_v)k_u = 0, ..., n_u$$
aux paramètres $u = \frac{k_u}{n_u}$,
- les distances curvilignes le long des courbes $\left\{C^v_{k_v}(u), k_v = 0, ..., n_v\right\}$ entre capteurs adjacents (k_u, k_v) et $(k_u + 1, k_v)$ sont de $L^v_{k_v}(k_u), k_u = 0, ..., n_u - 1$.

• Nous cherchons le nouveau réseau qui doit satisfaire les mêmes données de distances curvilignes entre les capteurs, mais doit répondre aux nouvelles données tangentielles :

- les courbes $\left\{\widetilde{C^u_{k_u}}(v), k_u = 0, ..., n_u\right\}$ sont tangentes aux vecteurs
$$\widetilde{T^u}(k_u, k_v)k_v = 0, ..., n_v$$
aux paramètres $v = \frac{k_v}{n_v}$,
- les courbes $\left\{\widetilde{C^v_{k_v}}(u), k_v = 0, ..., n_v\right\}$ sont tangentes aux vecteurs
$$\widetilde{T^v}(k_u, k_v)k_u = 0, ..., n_u$$
aux paramètres $u = \frac{k_u}{n_u}$.

La méthode de reconstruction de ce réseau de courbes déformé va se dérouler en deux étapes : tout d'abord reconstuire les courbes qui ont des contraintes de longueur, puis créer les autres courbes à partir des précédentes.

A. CRÉATION DES COURBES DÉFORMÉES DONT LES RÉPARTITIONS DE CAPTEURS SONT CONNUES

Nous allons tout d'abord créer les courbes qui ont des contraintes de longueur, c'est-à-dire les courbes $\left\{\widetilde{C_0^u}(v)\right\}$ et $\left\{\widetilde{C_{k_v}^v}(u), k_v = 0, ..., n_v\right\}$. Nous créons tout d'abord $\widetilde{C_0^u}(v)$, nous avons ainsi les positions des points de départ des courbes isoparamétriques en v :

$$\widetilde{C_{k_v}^v}(0) = \widetilde{C_0^u}\left(\frac{k_v}{n_v}\right), k_v = 0, ..., n_v$$

et nous reconstruisons alors les courbes $\left\{\widetilde{C_{k_v}^v}(u), k_v = 0, ..., n_v\right\}$.

Les courbes peuvent être reconstruites indépendamment les unes des autres, car ne se croisent pas : nous utilisons la méthode vue lors de la déformation de courbe vue dans la section 2.1.3 en page 90, et que nous utilisions également pour les deux courbes de bord de la section précédente.

B. CRÉATION DES AUTRES COURBES DÉFORMÉES

Nous devons à présent reconstruire les courbes dans l'autre sens afin de créer un réseau de courbes fermé : nous connaissons les données tangentielles ainsi que les positions des capteurs (grâce à la création des courbes dans l'autre sens). Nous procédons exactement de la même façon que pour la création du réseau initial : nous cherchons les courbes dans la base des fonctions de Hermite, avec des coefficients que nous cherchons en minimisant l'énergie sur chaque morceau de courbe (ce qui revient à résoudre un système linéaire). Nous avions détaillé cette méthode dans la section 3.4.2 en page 126.

Nous obtenons alors un réseau de courbes fermé : nous pouvons appliquer les méthodes de remplissage de Coons afin de définir la surface solution.

4.2 RÉSULTATS

Nous allons ici simuler une surface en déformation. Nous ne simulons pas les déformations avec contraintes de longueur des courbes, donc à chaque pas de temps, nous devrons calculer comme données les données tangentielles et les données de répartition entre capteurs adjacents. Cependant, nos méthodes de résolution ne s'appuient pas sur les contraintes de conservation de longueurs, mais sur la connaissance des longueurs : cela ne nous pose alors pas de problème, nous pouvons même dire que si en versions réelles de prototypes, nous pouvons avoir la connaissance d'étirement du système, notre algorithme s'appliquerait sans problème.

A chaque pas de temps, nous allons alors simuler les données des capteurs (données tangentielles), et les répartitions des capteurs. Puis nous appliquerons nos deux méthodes

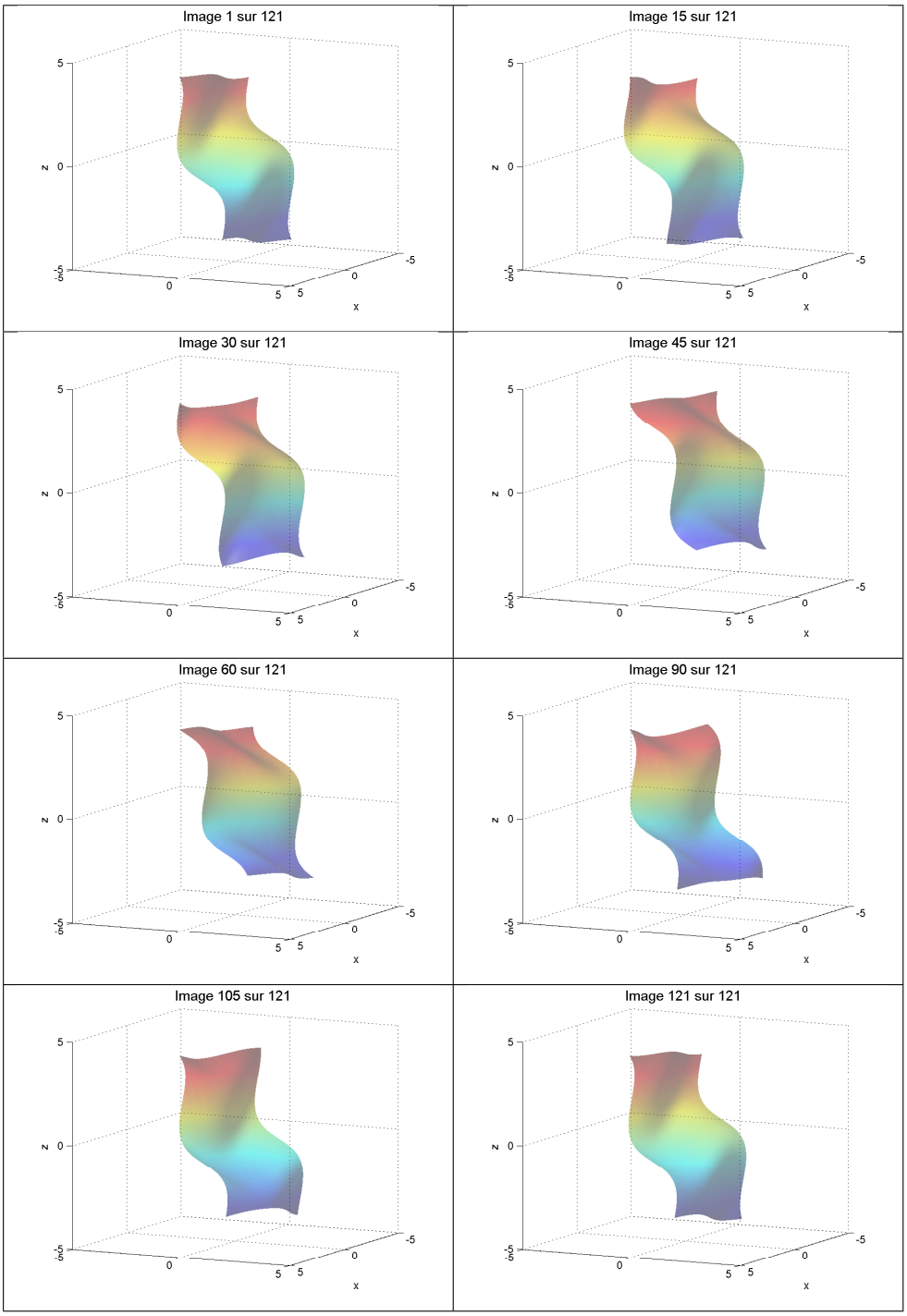
de reconstruction selon le système d'acquisition choisi (filet de capteurs, ou rubans instrumentés dans une seule direction). Nous calculerons les erreurs de reconstruction que nous avons déja explicitées dans le chapitre précédent : erreur de longueur moyenne des courbes reconstruites sur la surface, erreur de positions des capteurs, et erreur de Hausdorff.

4.2.1 TEST 1

Pour le premier test, nous déformons la surface des vagues où le paramètre t décrit la surface en fonction du temps (nous recalons le z afin de toujours avoir le point de départ fixe sur la surface) :

$$\text{pour} (u,v) \in [-4,4] \times [-2,2]$$
$$\text{pour } t = \frac{5k}{100}, k = 0, ..., 120.$$
$$\begin{cases} x_t(u,v) = u \\ y_t(u,v) = v \\ z_t(u,v) = \sin(u + t + \sin(v+t)) - \sin(-4 + \sin(-2)) + \sin(-4 + t + \sin(-2+t)) \end{cases}$$
puis rotation autour de y de $\frac{\pi}{3}$ puis rotation autour de z de $\frac{\pi}{3}$

Nous obtenons alors une séquence de 121 images. Voyons les évolutions de cette surface en quelques images dans le tableau TAB.4.1.



TAB. 4.1 – Déformation de la surface 1

Nous instrumentons cette surface par un réseau de 4 par 16 capteurs (voir FIG.4.1).

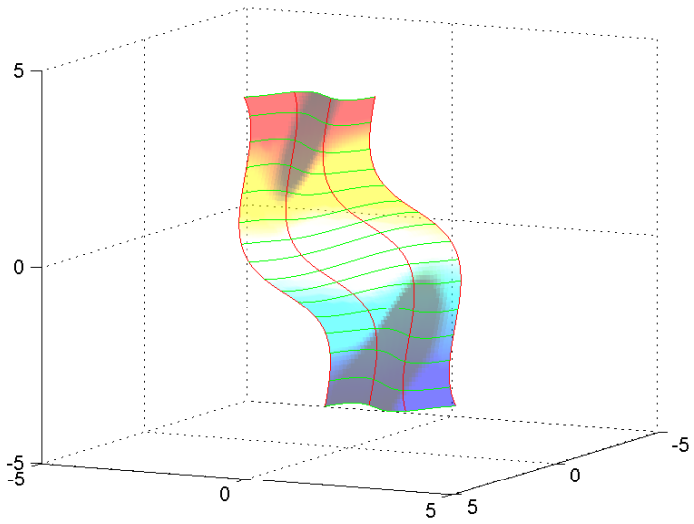


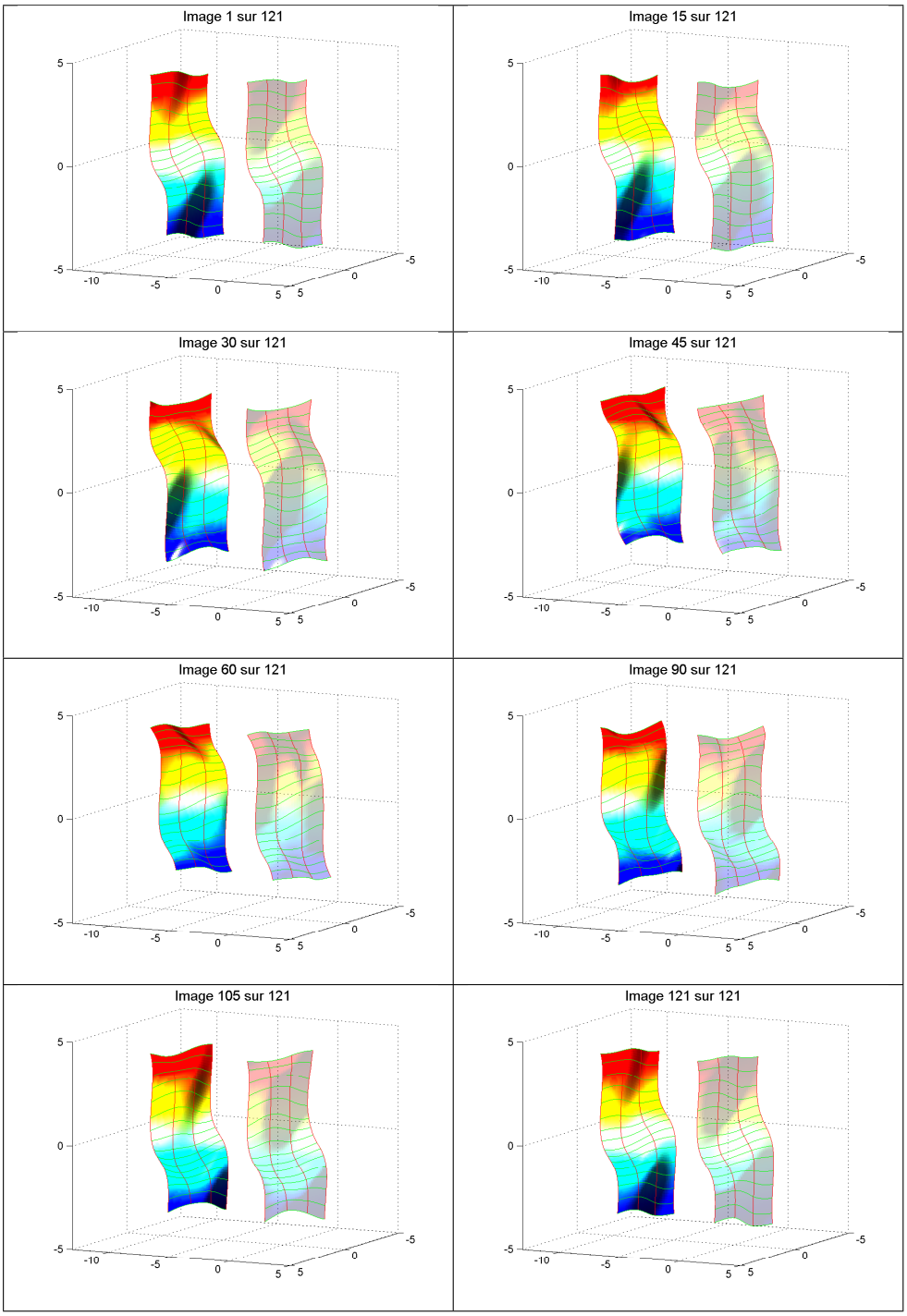
FIG. 4.1 – Instrumentation de la surface 1

Pour le premier système d'acquisition qui est le filet, nous avons alors les connaissances des longueurs inter-capteurs dans les deux directions. En ce qui concerne les rubans instrumentés, nous aurons 4 rubans instrumentés de 16 capteurs chacun.

Nous pouvons à présent appliquer nos méthodes de déformation.

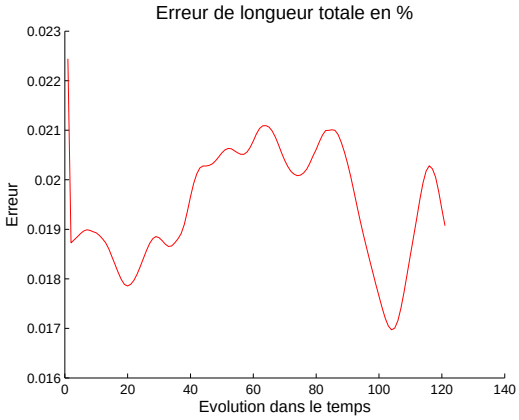
A. RECONSTRUCTION À PARTIR D'UN FILET DE DONNÉES

Nous appliquons pour la surface initiale la reconstruction avec la fermeture du réseau par contraintes de longueur, puis nous modifions le réseau en déformation avec la méthode détaillée dans ce chapitre où nous cherchons le nouveau réseau satisfaisant les contraintes tangentielles et de répartition des capteurs. Voici dans le tableau TAB.4.2 les résultats visuels obtenus pour les mêmes étapes que lors de la description de la surface en mouvement. Nous voyons sur la gauche la surface initiale avec son réseau de capteurs, et sur la droite la reconstruction avec son réseau reconstruit.

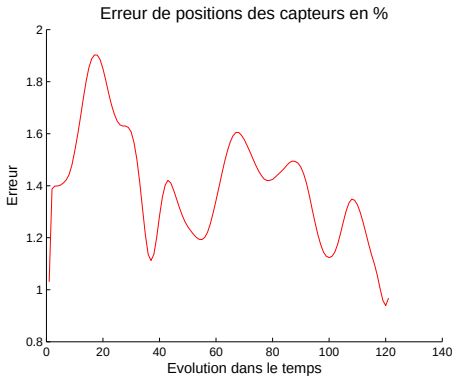


TAB. 4.2 – Suivi de déformation de la surface 1 grâce au filet de capteurs

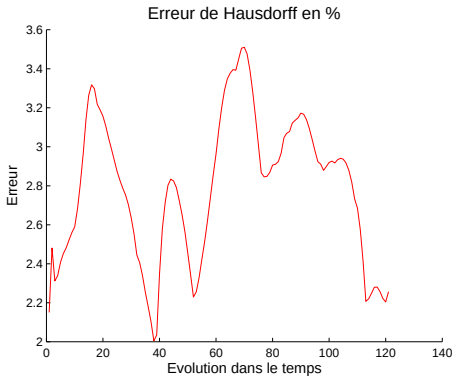
Les erreurs que nous avons pu calculer au cours du temps sont rassemblées dans la figure FIG.4.2.



(a) Erreur de longueur des courbes reconstruites



(b) Erreur de positions des capteurs



(c) Erreur de Hausdorff

FIG. 4.2 – Déformation test 1 : Erreurs de reconstruction avec le filet de capteurs

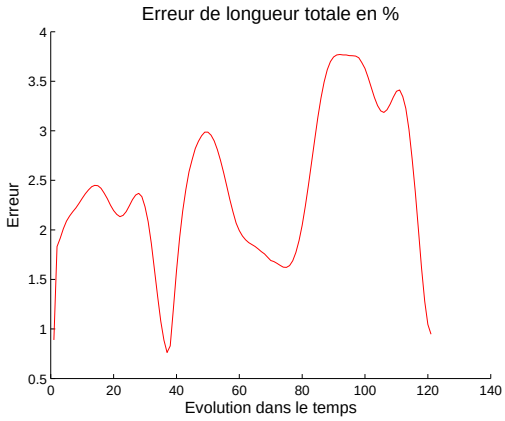
Nous pouvons déduire plusieurs éléments à partir de ces résultats. Le premier élément que nous voyons clairement est le fait que *les erreurs ne croissent pas en fonction du temps passé* : cela veut dire que la méthode de suivi de déformation ne diverge pas, nous n'aurons alors pas besoin de recalage avec la méthode initiale. De plus, les erreurs de longueur sont très faibles : la méthode respecte bien les contraintes de longueur. Enfin, en ce qui concerne les erreurs de positions des capteurs et les erreurs de Hausdorff, nous conservons de bons résultats tout au long de la déformation (les erreurs sont entre 1 et 2% pour la position des capteurs, et entre 2 et 3.5% pour l'erreur de Hausdorff).

Nous pouvons ainsi dire que la méthode développée donne de bons résultats lorsque nous possédons un filet de capteurs pour données.

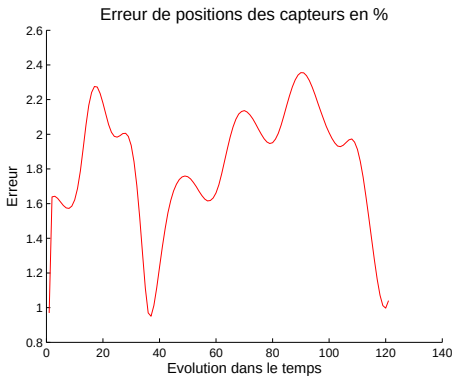
B. RECONSTRUCTION À PARTIR DE RUBANS INSTRUMENTÉS

Voyons à présent les reconstructions obtenues lorsque nous disposons d'une famille de rubans instrumentés dans la même direction. Nous visualisons les résultats pour les mêmes étapes que pour le test précédent (voir TAB.4.3).

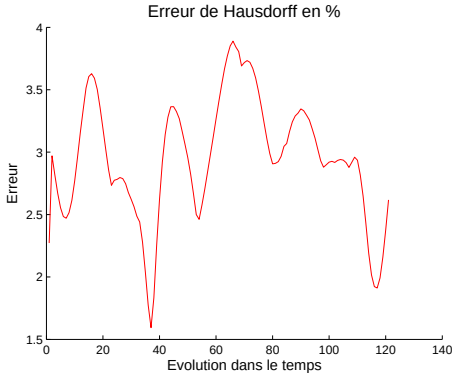
Nous faisons figurer dans la figure FIG.4.3 les différentes erreurs calculées au cours du temps.



(a) Erreur de longueur des courbes reconstruites

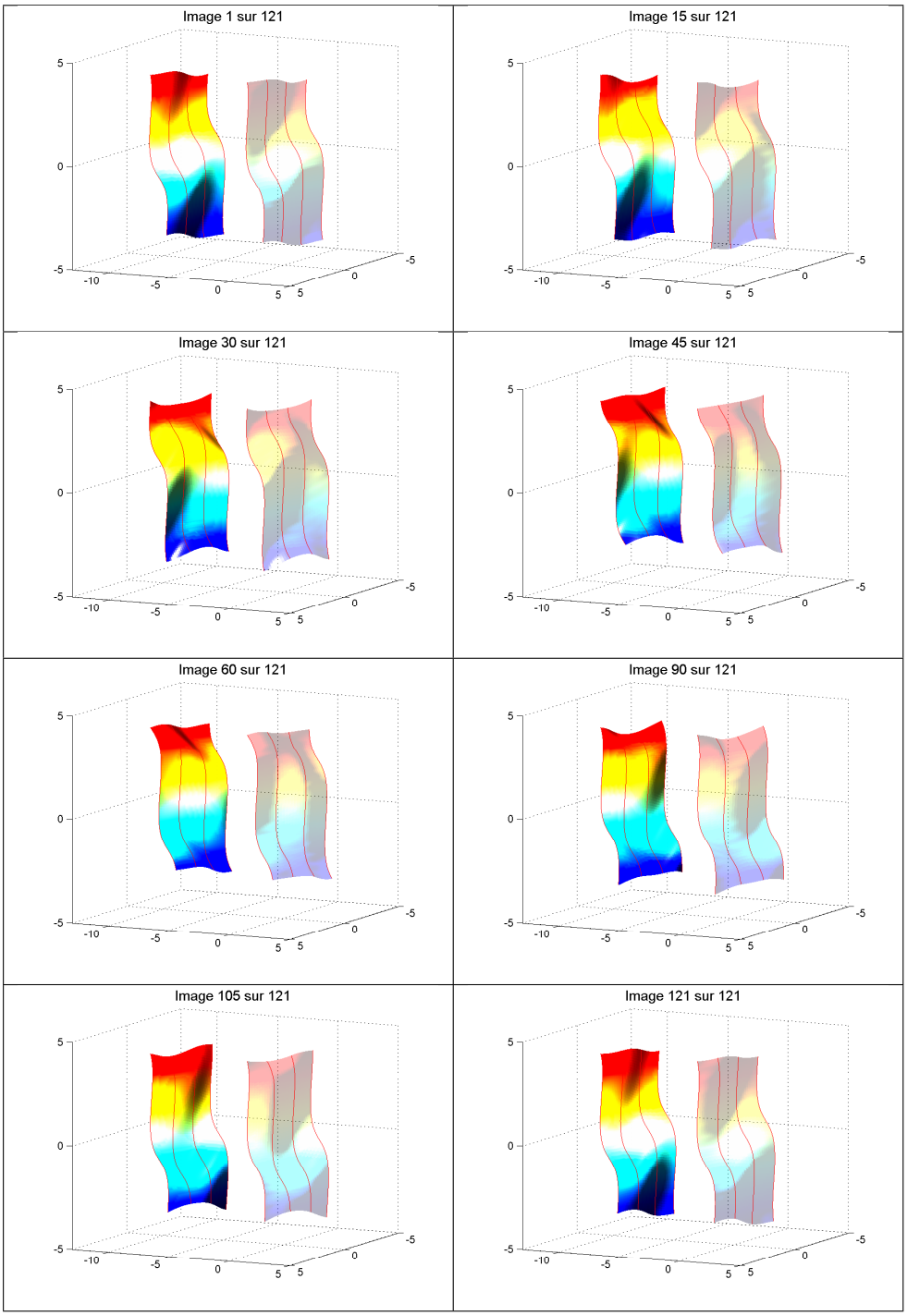


(b) Erreur de positions des capteurs



(c) Erreur de Hausdorff

FIG. 4.3 – Déformation test 1 : Erreurs de reconstruction avec les rubans instrumentés



TAB. 4.3 – Suivi de déformation de la surface 1 grâce aux rubans instrumentés

Ces résultats nous montrent également ici que la méthode ne diverge pas (nous obtenons les mêmes erreurs à la fin de la déformation qu'au début lorsque la surface a repris sa forme initiale). Les erreurs de longueur sont plus élevées que pour la méthode précédente, car la contrainte de longueur n'est à respecter que dans une direction, mais elles restent

acceptables (entre 1 et 4%). Les erreurs de positions des capteurs et les erreurs de Hausdorff sont quant à elles du même ordre que pour la méthode précédente de filet de données.

Donc nous pouvons aussi dire que cette méthode est satisfaisante sur cet exemple.

Voyons les résultats obtenus pour un deuxième exemple.

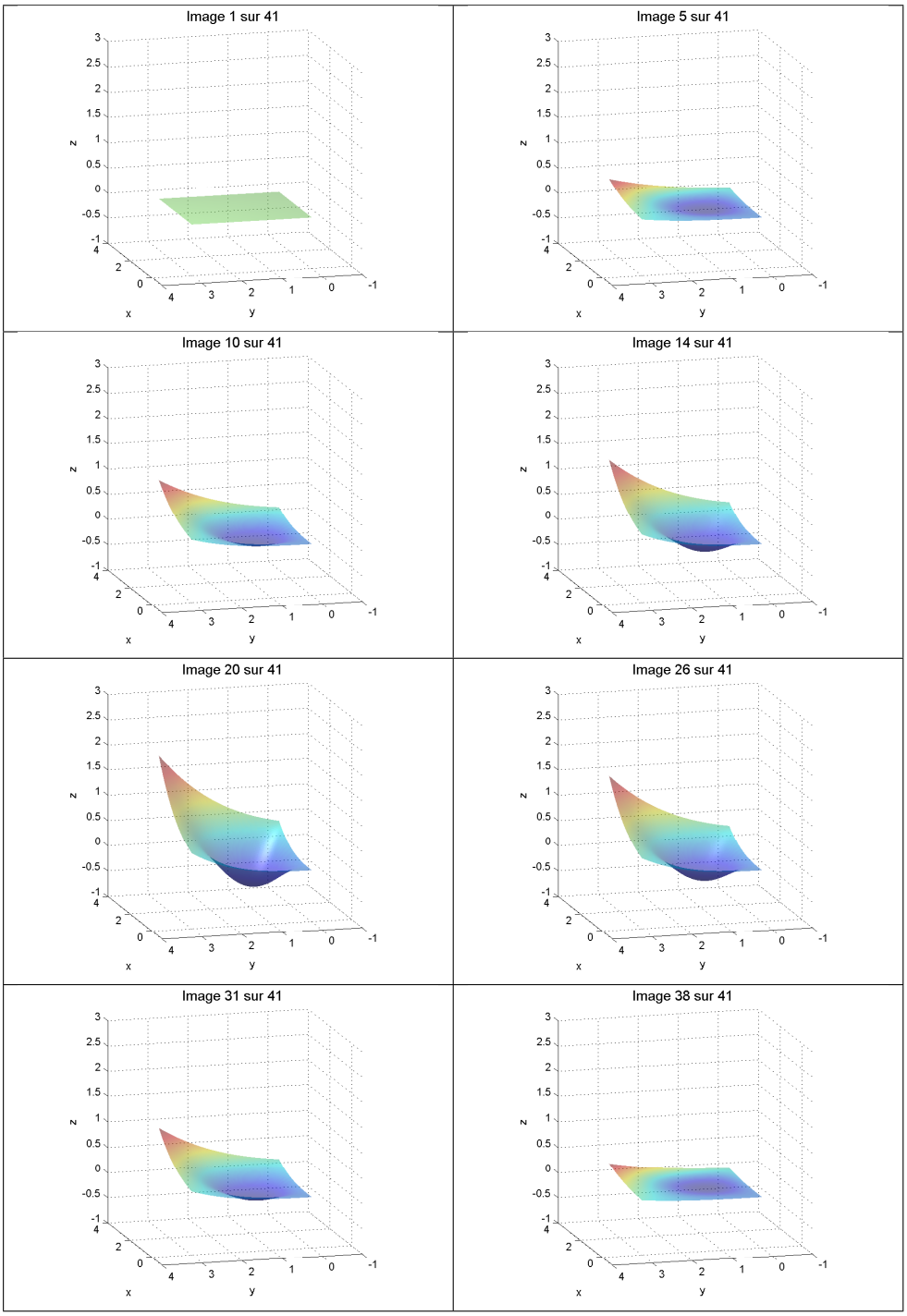
4.2.2 TEST 2

Pour ce deuxième test, nous utilisons une surface de Bézier, et nous la déformons en déplaçant certains points de contrôle. Les points de contrôle se déplacent en fonction du paramètre t de la façon suivante (nous déformons la surface puis la redéformons dans le sens inverse afin de retrouver la position initiale) :

Pour $t = \frac{k}{10}, k = 0, ..., 20$ puis $t = 2 - \frac{k}{10}, k = 1, ..., 20$,

$$\left\{ \begin{array}{llll} P_{00} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} & P_{01} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} & P_{02} = \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} & P_{03} = \begin{pmatrix} 3 \\ 0 \\ \frac{t}{4} \end{pmatrix} \\ P_{10} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} & P_{11} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} & P_{12} = \begin{pmatrix} 2 \\ 1 + \frac{t}{4} \\ -2.t \end{pmatrix} & P_{13} = \begin{pmatrix} 3 \\ 1 \\ \frac{t}{3} \end{pmatrix} \\ P_{20} = \begin{pmatrix} 0 \\ 2 \\ 0 \end{pmatrix} & P_{21} = \begin{pmatrix} 1 \\ 2 \\ 0 \end{pmatrix} & P_{22} = \begin{pmatrix} 2 \\ 2 \\ 0 \end{pmatrix} & P_{23} = \begin{pmatrix} 3 \\ 2 \\ \frac{t}{2} \end{pmatrix} \\ P_{30} = \begin{pmatrix} 0 \\ 3 \\ \frac{t}{4} \end{pmatrix} & P_{31} = \begin{pmatrix} 1 \\ 3 \\ \frac{t}{3} \end{pmatrix} & P_{32} = \begin{pmatrix} 2 \\ 3 \\ \frac{t}{2} \end{pmatrix} & P_{33} = \begin{pmatrix} 3 \\ 3 \\ t \end{pmatrix} \end{array} \right\}$$

Nous obtenons alors une séquence de 41 images. Voyons les évolutions de cette surface en quelques images dans le tableau TAB.4.4.



TAB. 4.4 – Déformation de la surface 2

Nous instrumentons cette surface par un réseau de 8 par 8 capteurs (voir FIG.4.4). Voyons les résultats obtenus pour les deux méthodes de calcul.

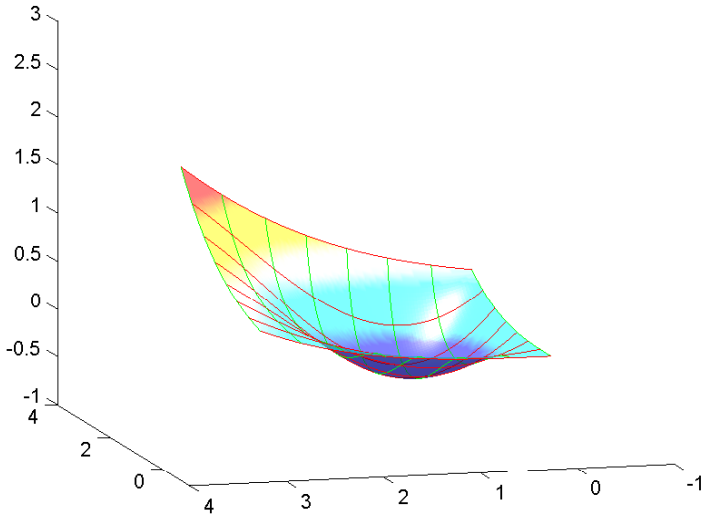
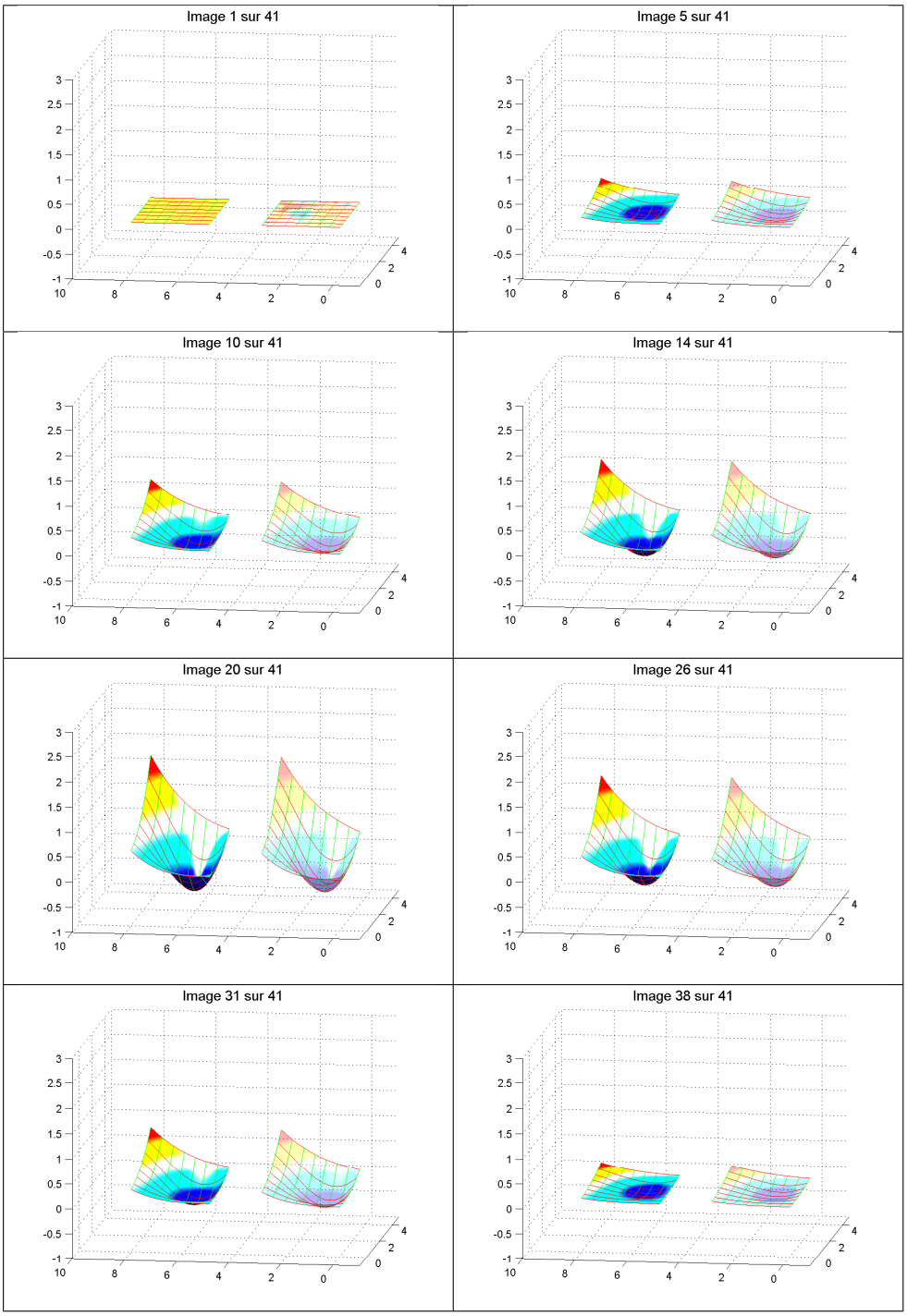


FIG. 4.4 – Instrumentation de la surface 2

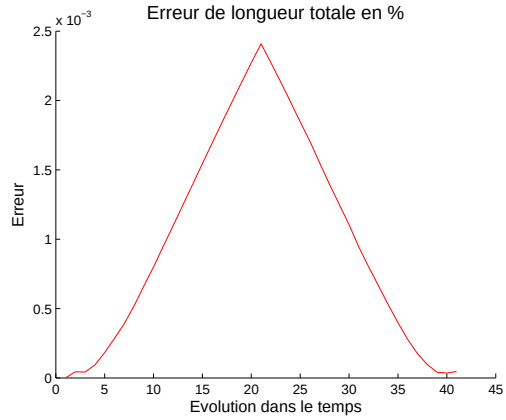
A. RECONSTRUCTION À PARTIR D'UN FILET DE DONNÉES

Appliquons ici la reconstruction de déformations en supposant que nous avons un filet de données de capteurs, c'est-à-dire où nous avons les données tangentielles ainsi que les répartitions curvilignes entre les capteurs adjacents. Voici dans le tableau TAB.4.5 les résultats visuels obtenus pour les mêmes étapes que lors de la description de la surface en mouvement. Nous voyons sur la gauche la surface initiale avec son réseau de capteurs, et sur la droite la reconstruction avec son réseau reconstruit.

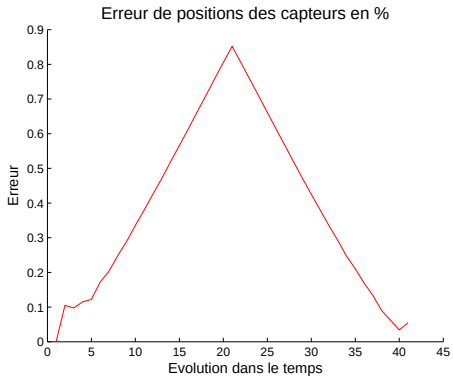


TAB. 4.5 – Suivi de déformation de la surface 2 grâce au filet de capteurs

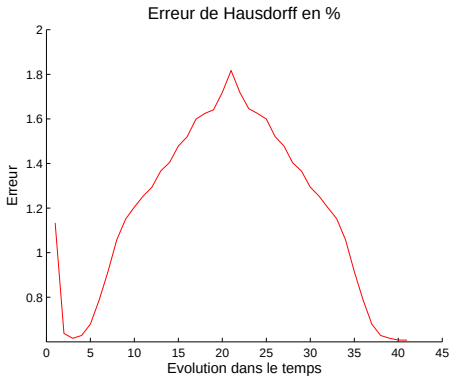
Les erreurs que nous avons pu calculer au cours du temps sont rassemblées dans la figure FIG.4.5.



(a) Erreur de longueur des courbes reconstruites



(b) Erreur de positions des capteurs



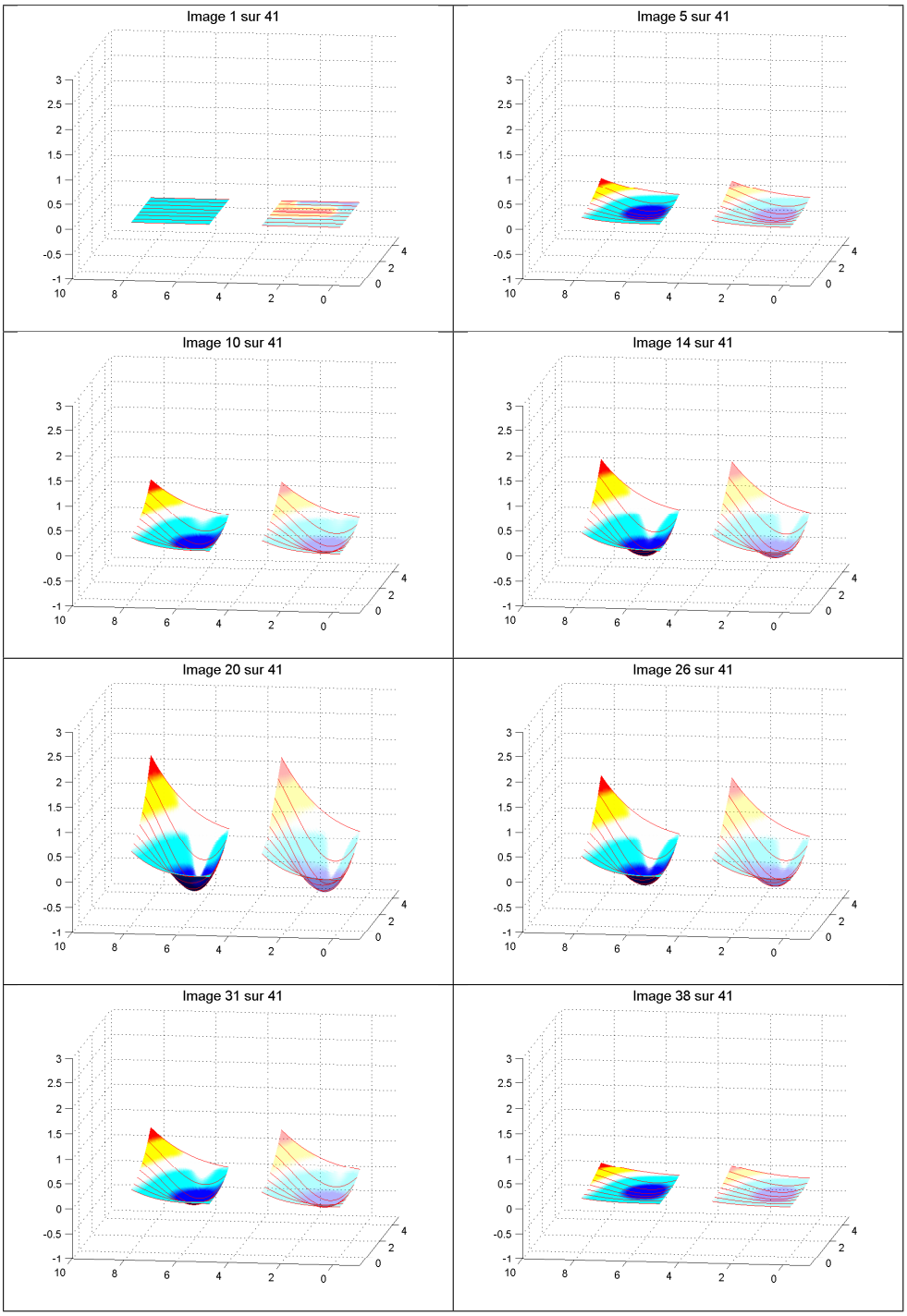
(c) Erreur de Hausdorff

FIG. 4.5 – Déformation test 2 : Erreurs de reconstruction avec le filet de capteurs

Ces courbes d'erreur sont symétriques : cela montre que les *erreurs dépendent de la forme de la surface et non de la façon dont on l' a obtenue.* La méthode de suivi de déformation ne diverge pas en fonction du temps sur cet exemple. De plus, nous voyons que les erreurs obtenues sont très faibles (de l'ordre de 10^{-3} pour les erreurs de longueur, moins de 1% d'erreur pour la position des capteurs, et moins de 2% d'erreur de Hausdorff).

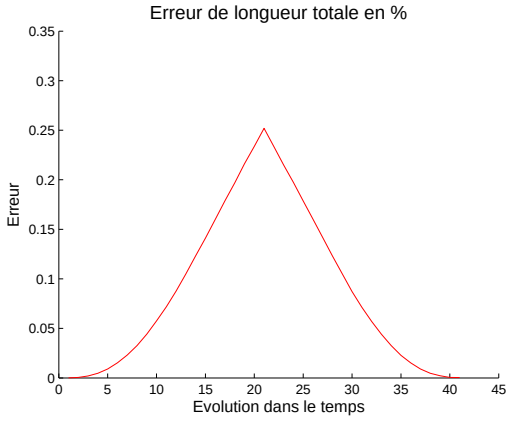
B. RECONSTRUCTION À PARTIR DE RUBANS INSTRUMENTÉS

Voyons à présent les reconstructions obtenues lorsque nous disposons d'une famille de rubans instrumentés dans la même direction. Nous visualisons les résultats pour les mêmes étapes que pour le test précédent (voir TAB.4.6).

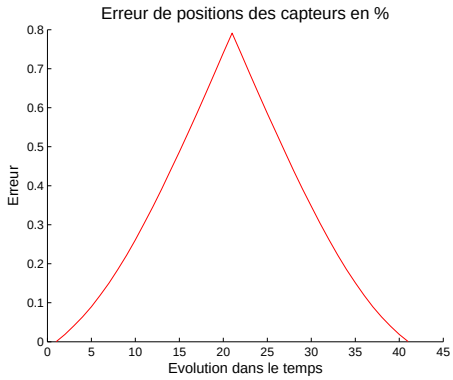


TAB. 4.6 – Suivi de déformation de la surface 2 grâce aux rubans instrumentés

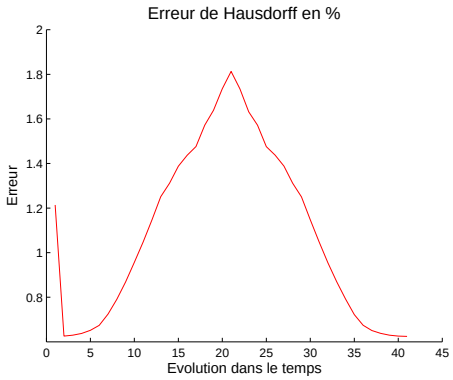
Nous faisons figurer dans la figure FIG.4.6 les différentes erreurs calculées au cours du temps.



(a) Erreur de longueur des courbes reconstruites



(b) Erreur de positions des capteurs



(c) Erreur de Hausdorff

FIG. 4.6 – Déformation test 2 : Erreurs de reconstruction avec les rubans instrumentés

Nous obtenons avec cette méthode des résultats similaires à ceux obtenus par la méthode du filet de capteurs : les résultats sont très satisfaisants.

4.3 CONCLUSION

Nous avons consacré cette partie au suivi de déformation de surface dans le temps. Nous avons ainsi pu ajouter aux contraintes données par les systèmes de capture de forme la donnée de la surface au moment précédent en sachant que la nouvelle surface était proche de la précédente. Nous avions au préalable développé des outils de suivi de déformation de courbes, ainsi que de reconstruction de réseaux de courbes sous contraintes de longueurs inter-capteurs.

Ces méthodes se déroulent en deux étapes : nous déformons tout d'abord le réseau de courbes, puis nous créons la surface qui s'appuie sur ces courbes avec les méthodes de remplissage de Coons. Pour la déformation du réseau de courbes, nous avons développé deux

différentes méthodes (selon le système de capture de forme dont nous pouvons disposer, soit un filet de capteurs, soit une famille de rubans de capteurs), méthodes qui réutilisent les différentes briques mises en œuvre dans les chapitres précédents.

En effet, si nous avons un filet de capteurs, nous reconstruisons les deux courbes de bord selon les méthodes de déformation de courbes, puis nous créons un réseau de courbes fermé sous contraintes de longueur proches du réseau initial. Et si nous disposons d'une famille de rubans instrumentés, nous reconstuisons les rubans instrumentés par la méthode de déformation de courbes, puis nous créons les courbes de l'autre direction exactement de la même manière que pour obtenir la surface initiale.

Nous avons testé ces méthodes sur deux exemples, sur lesquels nous avons pu juger de leur efficacité, avec des erreurs qui ne divergeaient pas d'une part, montrant que les méthodes n'avaient pas besoin de recalage, et des erreurs très faibles d'autre part, montrant ici que nous pouvions obtenir de très bonnes surfaces résulats.

Jusqu'à présent, nous avons développé les outils de reconstruction de courbes et de surfaces, tant en statique qu'en déformation, et nous les avons caractérisé sur des exemples simulés. Nous allons dans la dernière partie de ce document nous orienter sur le côté technologique de cette thèse, à savoir quels sont les outils technologiques qui nous permettent d'avoir des données tangentielles de courbes ou de surfaces, comment les utiliser, quels sont les prototypes qui ont été développés, et enfin quels résultats de reconstruction nous obtenons avec nos algorithmes.

Troisième partie

Les réalisations technologiques

Chapitre 5

Réalisations technologiques

Ce chapitre est consacré aux réalisations faites pour utiliser et tester les algorithmes que nous avons développés dans ce manuscrit. Nous détaillerons dans un premier temps les divers capteurs que nous pouvons utiliser et leur association afin de connaître les données tangentielles dont nous avons besoin pour reconstruire les courbes ou les surfaces sur lesquelles nous disposons les capteurs. Puis, nous verrons le ruban qui a été réalisé comme démonstrateur. Enfin, nous observerons les résultats de reconstruction obtenus avec ce prototype pour la reconstruction tant des courbes planes et gauches que des surfaces.

Sommaire

Ce chapitre montre les applications technologiques du travail de reconstruction de courbes et surfaces à partir de données tangentielles que nous avons développé tout au long de ce manuscrit. Nous avons réfléchi à l'élaboration d'un prototype capable de fournir de telles informations. Nous allons voir notre démarche et l'élaboration du démonstrateur réalisé.

5.1 DESCRIPTION DU PROTOTYPE

Voyons dans un premier temps comment il nous est possible d'obtenir des données tangentielles à partir de microcapteurs.

5.1.1 LES CAPTEURS

Le LETI conçoit depuis de nombreuses années des capteurs de données terrestres, qui sont ainsi capables de se géoréférencer. Deux sortes de capteurs sont développés. Nous avons d'une part les micro-accéléromètres, qui sont des capteurs sensibles à l'accélération, ils fournissent ainsi la direction du champ gravitationnel lorsque nous les utilisons de façon statique : nous avons alors la connaissance de la pente par rapport à la verticale de ce capteur. Nous avons d'autre part la possibilité d'utiliser des micro-magnétomètres, qui donnent des informations d'orientation par rapport au champ magnétique terrestre (lorsque aucune autre source magnétique n'est proche du capteur).

Ces capteurs donnent des valeurs dans un intervalle de données : la valeur la plus élevée étant lorsqu'il est orienté exactement dans le sens du champ qu'il mesure (par exemple, un accéléromètre immobile donnera sa valeur maximale lorsqu'il sera orienté verticalement vers le bas), et la valeur minimale est obtenue lorsqu'il est dans la direction opposée au champ qu'il mesure (pour l'accéléromètre, cela se produit donc lorsqu'il est en position verticale vers le haut). Ainsi, en rééchelonnant les valeurs entre 1 et -1, nous pouvons voir les capteurs comme donnant les valeurs de cosinus des angles qu'ils mesurent entre leur propre orientation et la direction du champ qu'ils mesurent.

Chaque capteur (accéléromètre ou magnétomètre) existe sous forme monoaxiale (un capteur dans une seule direction nous donnant ainsi l'angle qu'il fait par rapport à son champ de référence), sous forme biaxiale (deux capteurs orthogonaux entre eux donnent ainsi deux valeurs capteurs), ou sous forme triaxiale (nous avons un trièdre orthogonal de données capteurs).

Nous devons combiner plusieurs de ces capteurs pour avoir les données tangentielles dont nous avons besoin. Nous allons voir desquels nous devons nous servir dans la suite.

5.1.2 COMMENT OBTENIR LES DONNÉES TANGENTIELLES VOULUES

Le problème ici est de déterminer combien de capteurs nous allons mettre en chaque point de mesure afin d'avoir les informations tangentielles dont nous avons besoin (mono-axe, biaxe ou triaxe, accéléromètres et/ou magnétomètres). Nous allons supposer ici que les capteurs d'un même point de mesure sont posés exactement au même endroit.

Notre prototype a la forme d'un ruban, nous aurons alors à reconstruire la courbe qui forme l'axe du ruban. Ainsi, lorsque nous équipons notre système de capteurs, nous aurons accès à des informations plus importantes (à savoir l'information sur les plans tangents) que si nous avions seulement un fil instrumenté.

Pour reconstruire une courbe (qui est décrite par un ruban), nous avons besoin de connaître le vecteur tangent en chaque point de mesure (vecteur T dans la figure FIG.5.1).

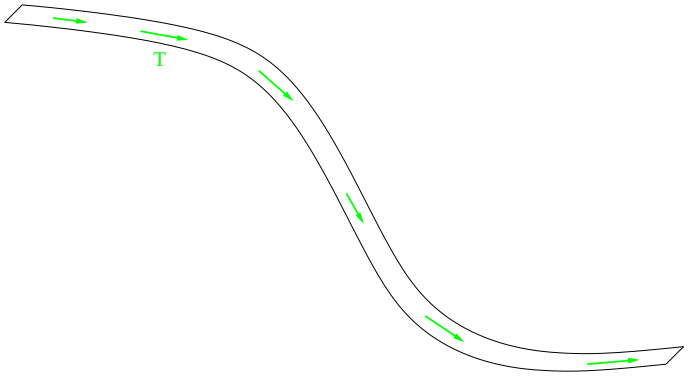


FIG. 5.1 – Ruban et données à extraire

Nous allons étudier le problème pour les deux cas suivants : tout d'abord si le ruban reste dans un plan vertical (il ne décrit alors qu'une courbe plane), puis lorsqu'il décrit une courbe dans l'espace.

A. RECONSTRUCTION DES COURBES PLANES

Nous supposons dans cette partie que le ruban décrit une courbe plane dans un plan vertical. Prenons alors un référentiel absolu bi-axe (e_1, e_2) avec e_1 en direction horizontale (par exemple en direction du nord) et e_2 vers le haut.

Les accéléromètres vont donner leur orientation par rapport au champ gravitationnel terrestre $A = \begin{pmatrix} 0 \\ -1 \end{pmatrix}$, et les magnétomètres donnent leur orientation par rapport au champ magnétique qui est orienté vers le nord avec un axe de 60° vers le bas (cette direction correspond au vecteur tangent aux lignes de champ passant au niveau du 45-ième parallèle), donc s'écrit dans la base $M = \begin{pmatrix} \frac{1}{2} \\ -\frac{\sqrt{3}}{2} \end{pmatrix}$.

Les vecteurs tangents cherchés sont des vecteurs unitaires en 2 dimensions, donc sont fonction d'un angle :

$$T = \begin{pmatrix} x = \cos\alpha \\ y = \sin\alpha \end{pmatrix}.$$

Nous devons ainsi déterminer l'angle α, voyons de combien de capteurs nous avons besoin.

Données d'un accéléromètre monoaxe Si nous plaçons un accéléromètre dans l'axe du vecteur tangent, il va nous donner l'angle γ_T^a, angle donné par l'accéléromètre (a) placé de long de l'axe tangent (T) (voir FIG.5.2).

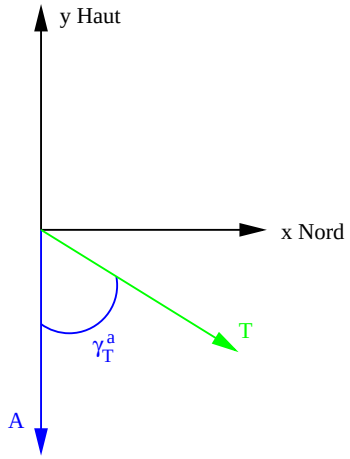


FIG. 5.2 – Donnée de l'accéléromètre monoaxe

Nous obtenons la relation suivante (le capteur fournit le résultat du produit scalaire entre les vecteurs A et T) :

$$\begin{aligned} T.\begin{pmatrix} 0 \\ -1 \end{pmatrix} &= \cos(\gamma_T^a) \\ -\sin\alpha &= \cos(\gamma_T^a) \\ \alpha &= \pm\gamma_T^a - \frac{\pi}{2} \end{aligned}$$

Nous voyons apparaître une indétermination où deux choix sont possibles : en effet, nous ne savons pas de quel coté nous sommes par rapport à la verticale. La figure FIG.5.3 illustre les deux vecteurs solutions (T et T_2) pour la donnée d'un même angle γ_T^a.

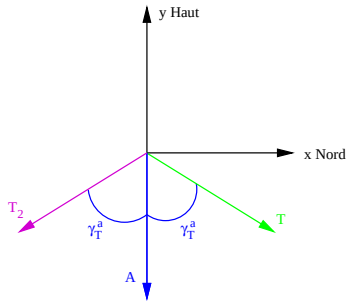


FIG. 5.3 – Ambiguité de solutions

Nous devons alors ajouter un autre capteur. Nous avons deux choix possibles : soit ajouter un magnétomètre dans l'axe tangent, soit ajouter un accéléromètre dans une direction orthogonale.

Données d'un accéléromètre monoaxe et d'un magnétomètre monoaxe Nous ajoutons un magnétomètre dans l'axe tangent. Nous ajoutons alors à la relation que nous avons grâce à l'accéléromètre la donnée du nouveau capteur γ_T^m (voir le schéma FIG.5.4), qui donne la relation suivante :

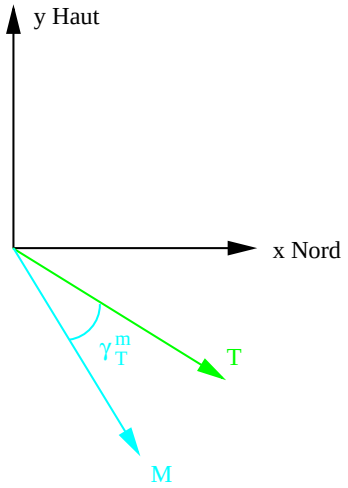


FIG. 5.4 – Donnée du magnétomètre monoaxe

$$\begin{aligned} T.\begin{pmatrix} \frac{1}{2} \\ -\frac{\sqrt{3}}{2} \end{pmatrix} &= \cos(\gamma_T^m) \\ \frac{1}{2}\cos\alpha - \frac{\sqrt{3}}{2}\sin\alpha &= \cos(\gamma_T^m) \\ \cos(\alpha + \frac{\pi}{3}) &= \cos(\gamma_T^m) \\ \alpha &= \pm\gamma_T^m - \frac{\pi}{3} \end{aligned}$$

Les deux capteurs nous donnent chacun 2 choix possibles qui sont différents : 1 seule solution sera commune, nous voyons son illustration sur la figure FIG.5.5, où pour la donnée de γ_T^a nous avons les deux solutions possibles T et T_2, et pour la donnée de γ_T^m, nous avons les deux solutions T et T_3, mais seul T est solution commune : c'est donc la solution du problème.

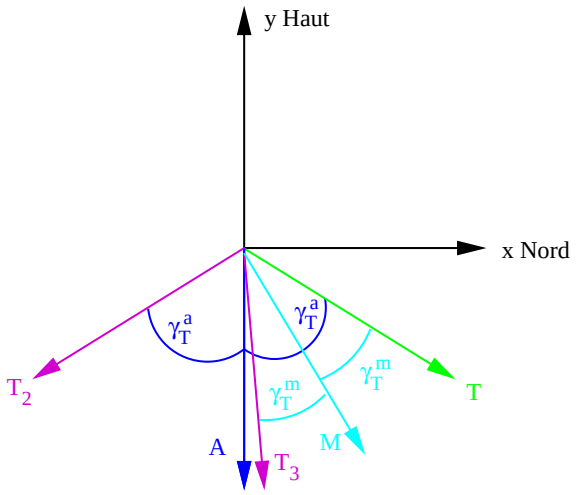


FIG. 5.5 – Informations accéléromètre et magnétomètre

Nous choisissons donc l'angle α de la façon suivante :

$$\text{si } \left|\gamma_T^a - \frac{\pi}{2} - \gamma_T^m + \frac{\pi}{3}\right| < \varepsilon \text{ ou } \left|\gamma_T^a - \frac{\pi}{2} + \gamma_T^m + \frac{\pi}{3}\right| < \varepsilon \text{ alors } \alpha = \gamma_T^a - \frac{\pi}{2}, \quad (5.1)$$
$$\text{sinon } \alpha = -\gamma_T^a - \frac{\pi}{2}.$$

Données d'un accéléromètre biaxe Voyons à présent comment nous pouvons connaître l'angle α à l'aide d'un accéléromètre biaxe. Nous avons en plus de la donnée de l'angle γ_T^a, la donnée d'un accéléromètre placé de manière orthogonale direct par rapport au vecteur T cherché. Si nous notons ce vecteur normal N, le capteur donne alors la valeur γ_N^a, qui est schématisé sur la figure FIG.5.6. Chaque capteur donne deux vecteurs solutions, donc

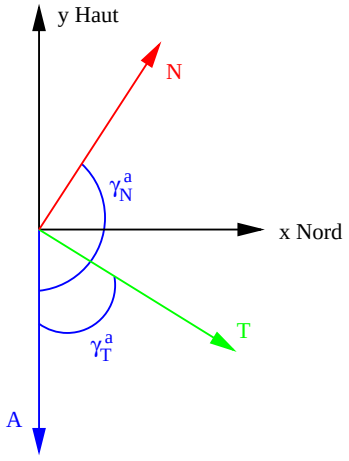


FIG. 5.6 – Données d'un accéléromètre biaxe

4 couples (T, N) solutions. Or, nous savons que le couple (T, N) constitue un repère direct, et parmi ces 4 couples solutions, 2 ne forment pas un repère orthogonal, et parmi les

deux couples solutions restants, un seul donne un repère direct : ceci nous permet de lever l'ambiguité.

Nous pouvons donc choisir la valeur de l'angle α en fonction de la donnée de $\cos(\gamma_N^a)$. En effet, nous voyons que si le vecteur T se situe dans les quadrants II et III (valeurs de x négatives) alors le vecteur N se situe dans les quadrants III et IV (valeurs y négatives donc $0 \leq \cos(\gamma_N^a)$). Et dans l'autre cas (T dans les quadrants IV et I), nous aurons $\cos(\gamma_N^a) \leq 0$. Ainsi seul le signe de $\cos(\gamma_N^a)$ suffit à déterminer la valeur de l'angle α recherché :

$$\begin{aligned} &\text{si } 0 \leq \cos(\gamma_N^a) \text{ alors } \alpha = \gamma_T^a - \frac{\pi}{2}, \\ &\text{sinon } \alpha = -\gamma_T^a - \frac{\pi}{2}. \end{aligned} \tag{5.2}$$

Conclusion Pour la connaissance totale de données tangentielles afin de reconstruire des courbes planes, nous avons besoin de deux valeurs capteurs en un point de mesure : soit un accéléromètre monoaxe dans la direction tangentielle combiné à un magnétomètre monoaxe dans cette même direction, soit un accéléromètre biaxe.

B. Reconstruction des courbes gauches

Voyons à présent quels capteurs sont nécessaires à la connaissance des vecteurs tangents pour la reconstruction de courbes gauches. Dans ce cas, nous devons déterminer les vecteurs tangents dans l'espace $3D$, vecteurs unitaires :

$$T = \begin{pmatrix} x \\ y \\ z \end{pmatrix}, x^2 + y^2 + z^2 = 1.$$

Nous nous plaçons dans le repère (e_1, e_2, e_3), avec e_1 vers le nord, e_2 vers l'ouest et e_3 vers le haut. Nous pouvons ainsi exprimer nos 2 vecteurs de référence : le vecteur représentant la direction du champ gravitationnel terrestre $A = \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}$ et le vecteur représentant la direction du champ magnétique terrestre : $M = \begin{pmatrix} \frac{1}{2} \\ 0 \\ -\frac{\sqrt{3}}{2} \end{pmatrix}$.

Nous allons décrire par la suite progressivement les capteurs que nous utilisons et les données qu'ils fournissent.

Donnée d'un accéléromètre monoaxe et d'un magnétomètre monoaxe Cela nous donne l'angle γ_T^a que fait le vecteur tangent et la verticale :

$$\begin{aligned} T.A &= \cos(\gamma_T^a) \\ z &= -\cos(\gamma_T^a) \end{aligned}$$

et l'angle γ_T^m que fait le vecteur tangent et la direction du champ magnétique :

$$\begin{aligned} T.M &= \cos(\gamma_T^m) \\ \frac{1}{2}x - \frac{\sqrt{3}}{2}z &= \cos(\gamma_T^m) \\ x &= 2\cos(\gamma_T^m) - \sqrt{3}\cos(\gamma_T^a) \end{aligned}$$

Nous connaissons parfaitement 2 coordonnées sur les 3 et nous savons que le vecteur est unitaire : il nous reste donc 2 possibilités pour la coordonnée y. Le vecteur est de la forme

$$\begin{aligned} z &= -\cos(\gamma_T^a) \\ x &= 2\cos(\gamma_T^m) - \sqrt{3}\cos(\gamma_T^a) \\ y &= \pm\sqrt{1 - x^2 - z^2} \end{aligned}$$

Les deux capteurs monoaxes ne suffisent pas : nous devons ajouter d'autres capteurs. Nous allons ajouter un accéléromètre et un magnétomètre dans une direction orthogonale au vecteur tangent.

Donnée d'un accéléromètre biaxe et d'un magnétomètre biaxe Nous allons les placer dans le plan du ruban, en direction orthogonale directe par rapport à ce plan du ruban (voir le schéma sur la figure FIG.5.7).

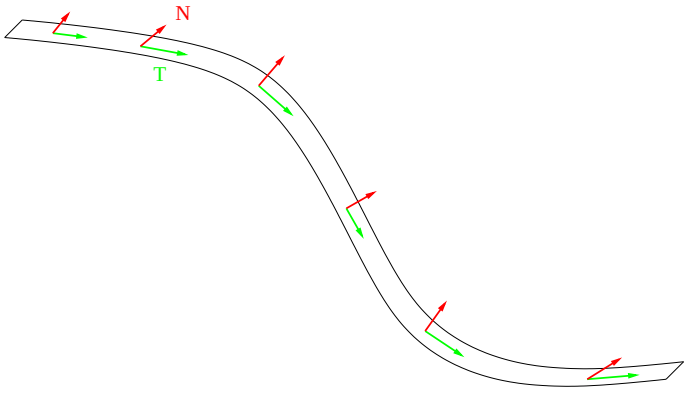


FIG. 5.7 – Ruban avec les capteurs biaxes

Les informations que nous aurons en plus ici concernent le vecteur N, qui peut nous aider à lever l'indétermination de T en utilisant le fait que T et N sont des vecteurs orthogonaux. Nous avons :

$$T = \begin{pmatrix} x \\ y \\ z \end{pmatrix}, N = \begin{pmatrix} x_N \\ y_N \\ z_N \end{pmatrix}$$

Les capteurs placés dans la direction de T nous donnent les renseignements vus dans la partie précédente, les deux autres capteurs nous donnent des informations similaires

concernant le vecteur N (avec les données capteurs γ_N^a et γ_N^m), et nous savons que les vecteurs sont orthogonaux :

$$T = \begin{pmatrix} x = 2\cos(\gamma_T^m) - \sqrt{3}\cos(\gamma_T^a) \\ y = \pm\sqrt{1-x^2-z^2} \\ z = -\cos(\gamma_T^a) \end{pmatrix},$$

$$N = \begin{pmatrix} x_N = 2\cos(\gamma_N^m) - \sqrt{3}\cos(\gamma_N^a) \\ y_N = \pm\sqrt{1-x_N^2-z_N^2} \\ z_N = -\cos(\gamma_N^a) \end{pmatrix},$$

$$x.x_N + y.y_N + z.z_N = 0$$

Nous avons ainsi deux choix pour T, deux pour N, soit en tout quatre couples possibles. Le fait que T et N soient orthogonaux va éliminer des possibilités.

Soit $T_1 = (x, y, z)$, et $N_1 = (x_N, y_N, z_N)$ orthogonaux, (nous avons la relation $x.x_N + y.y_N + z.z_N = 0$) et dans ce cas, nous avons $T_2 = (x, -y, z)$ et $N_2 = (x_N, -y_N, z_N)$ qui sont aussi orthogonaux (car $x.x_N + (-y).(-y_N) + z.z_N = x.x_N + y.y_N + z.z_N = 0$) : ces contraintes supplémentaires ne nous permettent pas de déterminer la solution : il y a toujours 2 solutions possibles :

si $y.y_N = -x.x_N - z.z_N > 0$ alors ($y > 0$ et $y_N > 0$) ou ($y < 0$ et $y_N < 0$)
sinon ($y > 0$ et $y_N < 0$) ou ($y < 0$ et $y_N > 0$)

Nous avons alors besoin de données supplémentaires : nous allons ajouter un capteur dans la direction orthogonale aux deux autres afin d'obtenir des informations sur les trois axes d'un trièdre direct.

Données d'un accéléromètre triaxe et d'un magnétomètre biaxe Les quatre capteurs utilisés ne permettent pas de déterminer le couple solution, il nous reste encore deux possibilités (les deux couples restants étant des vecteurs orthogonaux entre eux). Nous allons devoir trancher en ajoutant une information sur le troisième vecteur du trièdre. En créant le vecteur B, vecteur formant un trièdre direct avec T et N, nous pourrons éliminer la mauvaise solution (voir sur la figure FIG.5.8).

En effet, soit $(T_1 = (x, y, z), N_1 = (x_N, y_N, z_N))$ le premier couple solution. Dans ce cas, le vecteur B_1 du trièdre (T_1, N_1, B_1) s'écrit comme le produit vectoriel $B_1 = T_1 \wedge N_1$ et sa troisième coordonnée est alors :

$$z_{B_1} = x.y_N - y.x_N.$$

Le deuxième couple solution est $(T_2 = (x, -y, z), N_2 = (x_N, -y_N, z_N))$, et son vecteur B_2 associé a pour troisième coordonnée :

$$z_{B_2} = x.(-y_N) - (-y).x_N = -z_{B_1}.$$

Nous pouvons donc départager les deux trièdres solutions par la connaissance du signe de z_B : c'est pourquoi nous allons ajouter un accéléromètre dans la direction de B.

Le cinquième capteur ajouté donne la valeur γ_B^a, nous avons toutes les données suivantes :

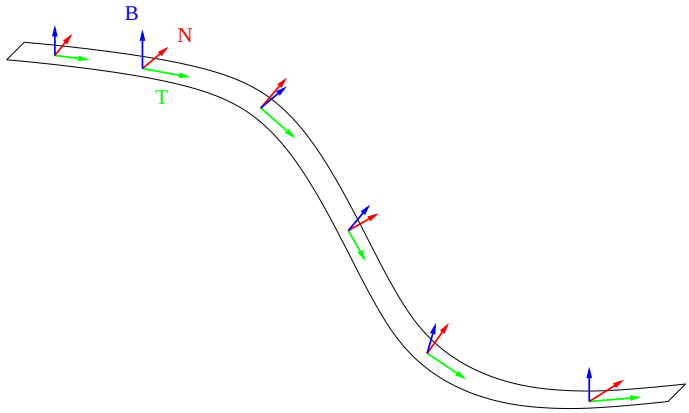


FIG. 5.8 – Trièdre de données

$$T = \begin{pmatrix} x = 2\cos(\gamma_T^m) - \sqrt{3}\cos(\gamma_T^a) \\ y = \pm\sqrt{1 - x^2 - z^2} \\ z = -\cos(\gamma_T^a) \end{pmatrix}, N = \begin{pmatrix} x_N = 2\cos(\gamma_N^m) - \sqrt{3}\cos(\gamma_N^a) \\ y_N = \pm\sqrt{1 - x_N^2 - z_N^2} \\ z_N = -\cos(\gamma_N^a) \end{pmatrix}$$

$$z_B = -\cos(\gamma_B^a)$$

$$x.x_N + y.y_N + z.z_N = 0$$

$$\det(T, N, B) = 1$$

Avec la contrainte de T et N orthogonaux, nous avons une connaissance des 2 solutions possibles : nous supposons $y > 0$:

$$\text{si } y.y_N = -x.x_N - z.z_N > 0 \text{ alors } y_N > 0 \text{ sinon } y_N < 0$$

Le produit vectoriel nous donne une relation entre z_B et les coordonnées de T et N

$$z_B = x_N.y - y_N.x$$

Ainsi nous calculons $x_N.y - y_N.x$. Si cela vaut z_B, alors notre hypothèse est correcte (nous avons bien $y > 0$), sinon (le calcul de la troisième coordonnée du produit vectoriel vaut alors $-z_B$) nous avons $y < 0$.

Conclusion Nous venons de voir que pour connaître parfaitement la donnée tangentielle en un point de mesure pour la reconstruction d'une courbe dans l'espace, nous avons besoin de 5 valeurs capteurs : des accéléromètres dans les trois axes du trièdre et des magnétomètres dans les deux axes du plan tangent au ruban. Cela nous permet en fait de connaître tout le trièdre associé au point de mesure.

5.1.3 LE RUBAN PROTOTYPE

Voyons à présent le prototype qui a été réalisé. Ce ruban souple nommé MorphoSense, d'une largeur environ de 3 centimètres, alterne 16 accéléromètres triaxes et 16 magnétomètres biaxes tous les 2,5 centimètres environ. Nous voyons sur la photo de la figure

FIG.5.9 les capteurs dans le ruban (un magnétomètre est repéré par un cercle rouge et un accéléromètre est repéré par un cercle vert).

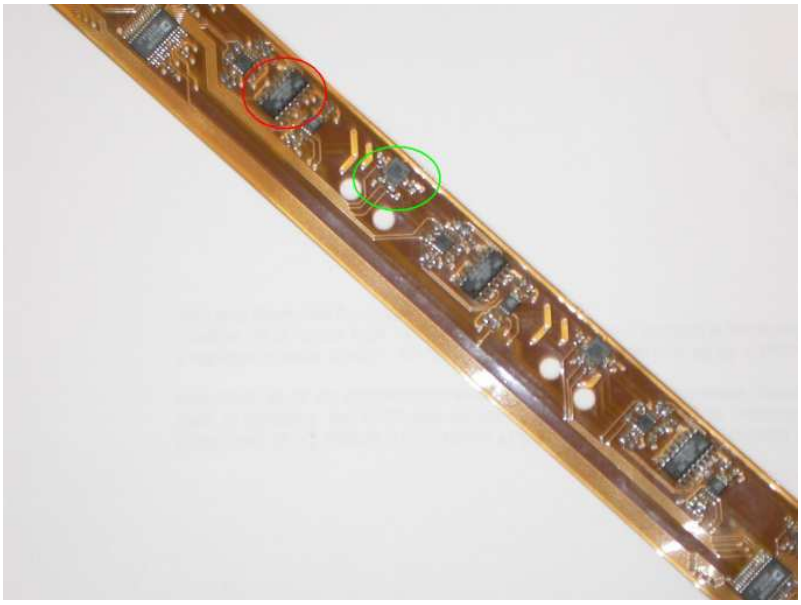


FIG. 5.9 – Ruban MorphoSense et ses capteurs

Nous pouvons également voir la version finale du ruban moulé (ruban qui a été couvert d'un matériau plastique large pour le protéger et lisser les mouvements) (voir photo sur la figure FIG.5.10).

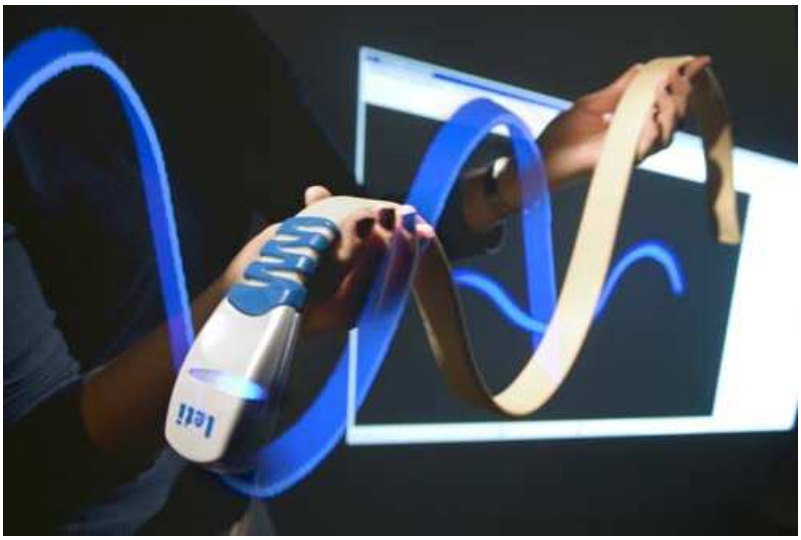


FIG. 5.10 – Ruban MorphoSense

Pour la reconstruction de courbes planes, nous allons nous servir de la donnée seule des accéléromètres (nous avons vu précédemment que la donnée d'un accéléromètre biaxe suffisait) : nous supposons alors que nous disposons d'un ruban de 16 points de mesure (où

nous utilisons seulement deux valeurs accélérométriques sur les trois et nous ignorons les magnétomètres), et les capteurs sont répartis tous les 5 centimètres.

Pour la reconstruction de courbes gauches, nous avons besoin des 5 données capteurs en chaque point de mesure. Or les 3 données accélérométriques sont en un point de mesure et les 2 données magnétométriques sont situées aux points de mesure voisins. Plusieurs possibilités sont envisagées. Nous avons choisi de créer 32 points de mesure sur la courbe à reconstruire, avec aux points de mesure accélérométrique les valeurs moyennées des valeurs magnétométriques voisines et le contraire aux vrais points de mesure magnétométrique. Cette tactique de reconstruction ne pose pas de problème dans le sens où les capteurs sont proches les uns des autres, et le matériau assez rigide pour que les valeurs moyennées soient proches des valeurs réelles en ces points voisins.

Afin d'utiliser ce ruban, nous devons le calibrer, c'est-à-dire déterminer les valeurs caractéristiques de chaque capteur pour les reétalonner entre -1 et 1 (et avoir ainsi les valeurs de cosinus d'angle dont nous avons parlé précédemment). Nous avons décrit les différentes étapes de la calibration dans l'annexe D.

5.1.4 Conclusion

La partie précédente nous a permis de déterminer technologiquement les possibilités pour appliquer les algorithmes de reconstruction de courbes à partir de données tangentielles. Nous avons déterminé les capteurs dont nous avions besoin, les outils pour obtenir les données tangentielles (calibration des capteurs, et formules analytiques pour avoir réellement les données tangentielles), et nous avons décrit le ruban prototype qui a été créé à partir de ces réflexions.

Nous pouvons à présent utiliser MorphoSense pour décrire les courbes dans l'espace. Nous allons par la suite voir les résultats que nous avons obtenus.

5.2 Les résultats obtenus

5.2.1 Les courbes

A. Caractérisation du ruban

Afin de tester la validité du prototype utilisé, nous avons fabriqué un gabarit ayant la forme d'une sinusoïde. Ainsi, nous pouvons comparer le résultat de reconstruction du prototype par la méthode que nous avons développée avec la courbe réelle que ce gabarit représente (Fig.5.11).

FIG. 5.11 – Gabarit sinusoïdal

Nous avons posé le ruban durant plusieurs secondes sur le gabarit, le ruban prenant des mesures de façon continue, nous avons ainsi extrait assez d'informations pour reconstruire 840 courbes. Nous avons dans un premier temps reconstruit une courbe à partir des données moyennées (voir figure FIG.5.12).

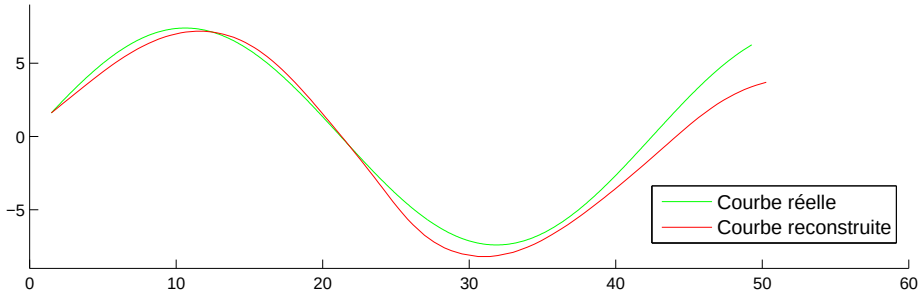


FIG. 5.12 – Sinusoïde reconstruite par MorphoSense

Cette courbe moyennée admet une erreur de longueur de $2.69.10^{-2}\%$, une erreur de position globale de 1.64% et une erreur de Hausdorff de 4.52%.

Nous avons également reconstruit les 840 courbes à partir des données, où nous avons calculé ces trois erreurs, nous les voyons dans la figure FIG.5.13.

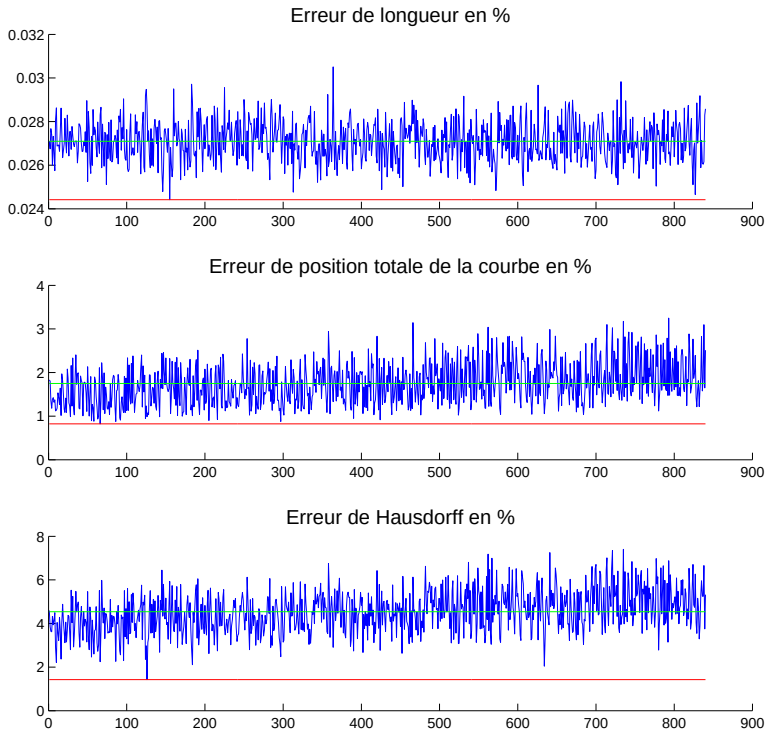


FIG. 5.13 – Erreurs de reconstruction du gabarit 2D

Nous y avons aussi représenté les valeurs minimales en rouge et les valeurs moyennes en vert. Nous voyons ainsi que les courbes obtenues ont toutes de très faibles erreurs de longueur (une moyenne de 0.024%), ainsi que des erreurs assez faibles pour la position globale (1.7% de moyenne) et l'erreur de Hausdorff (4% de moyenne).

Ces résultats montrent que nous obtenons de bons résultats malgré des données réelles de capteurs (dont les valeurs sont précises à 1° près). La méthode de reconstruction de courbes planes est donc adaptée à ce système de capture de forme.

Ces erreurs sont acceptables pour certaines applications mais nous pouvons avoir besoin d'être plus précis dans d'autres cas. Des améliorations sont alors à envisager pour le système : augmenter la densité des capteurs est une possibilité.

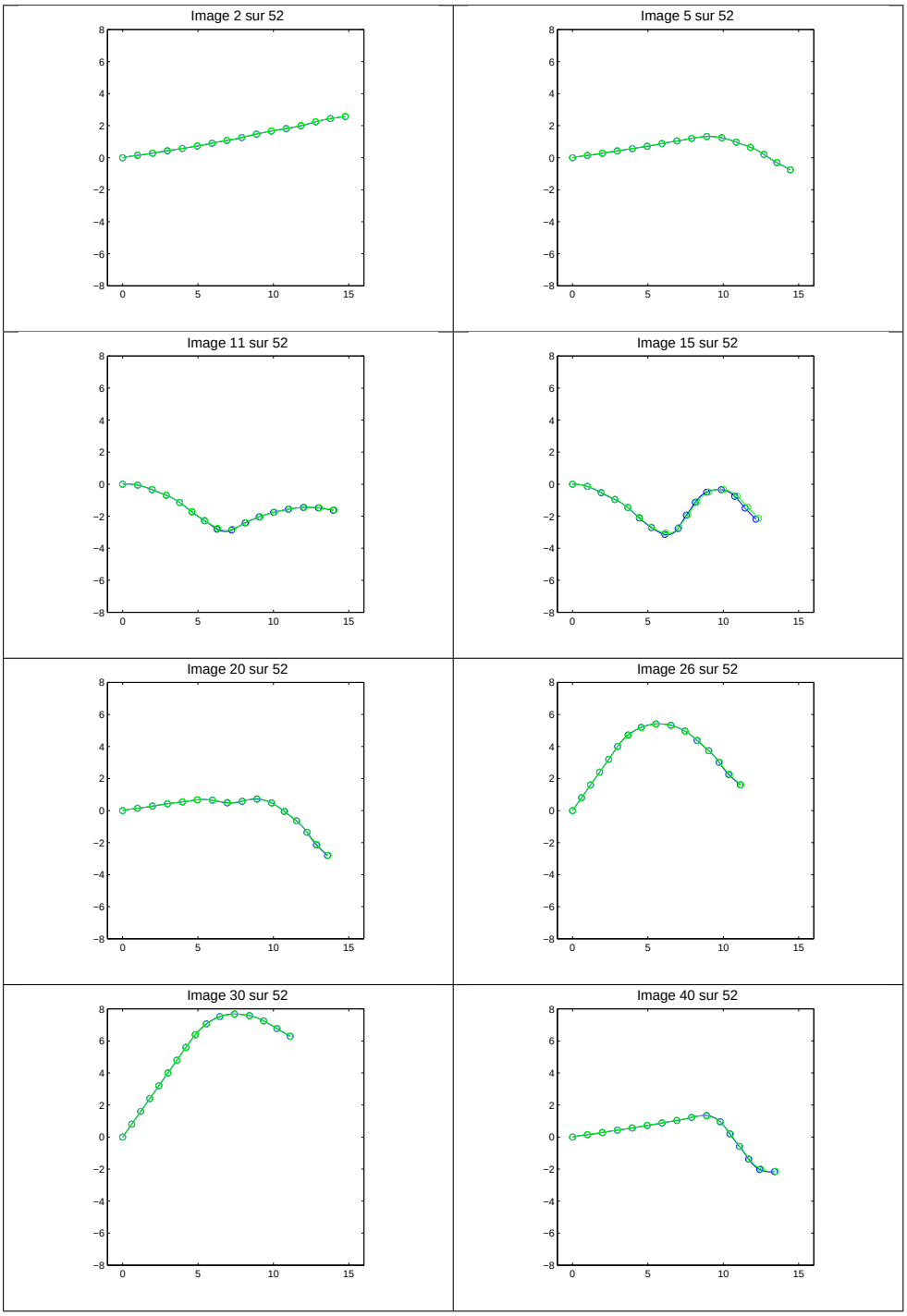
B. Capture de mouvement de courbes

Voyons à présent les résultats que nous obtenons lors du suivi de déformations de courbes. Nous allons décrire des mouvements avec le ruban. Nous devrons alors suivre l'évolution de sa forme au cours du temps. Nous rappelons que nous devons faire des mouvements n'ayant aucune accélération propre, car les capteurs accéléromètres fourniraient alors non plus des informations sur leur orientation, mais une information donnant aussi l'accélération du mouvement : les mouvements que nous faisons avec le ruban sont alors des mouvements assez lents.

Nous allons comparer les deux méthodes que nous avons développées : soit nous allons reconstruire à chaque pas de temps la courbe comme si nous ne connaissions pas sa forme au temps précédent, soit nous allons utiliser la méthode de déformation de courbes. Nous allons tout d'abord voir les résultats pour divers exemples de courbes planes, puis nous verrons des résultats sur les courbes gauches.

Courbe plane

Test 1 Pour le premier test, nous déformons le ruban de façon à créer une suite de 52 images. La courbe initiale est créée à partir de l'algorithme de base (à partir des données tangentielles seulement et avec la méthode d'interpolation des angles). Nous utilisons ainsi 51 fois l'algorithme de déformation de courbes. Nous allons voir ici des comparaisons visuelles : dans le tableau Tab.5.1 nous voyons différentes étapes de la déformation, où la courbe bleue est celle obtenue avec la méthode d'intégration, et la verte est obtenue avec la méthode de suivi de déformation.



TAB. 5.1 – Test 1 de déformation de courbes planes avec MorphoSense

Nous obtenons visuellement deux familles de courbes très proches les unes des autres. Afin de comparer ces deux méthodes, nous avons calculé les erreurs de longueur, de positions de capteurs, de position globale et de Hausdorff d'une méthode par rapport à l'autre. Ces résultats sont rassemblés dans la figure FIG.5.14.

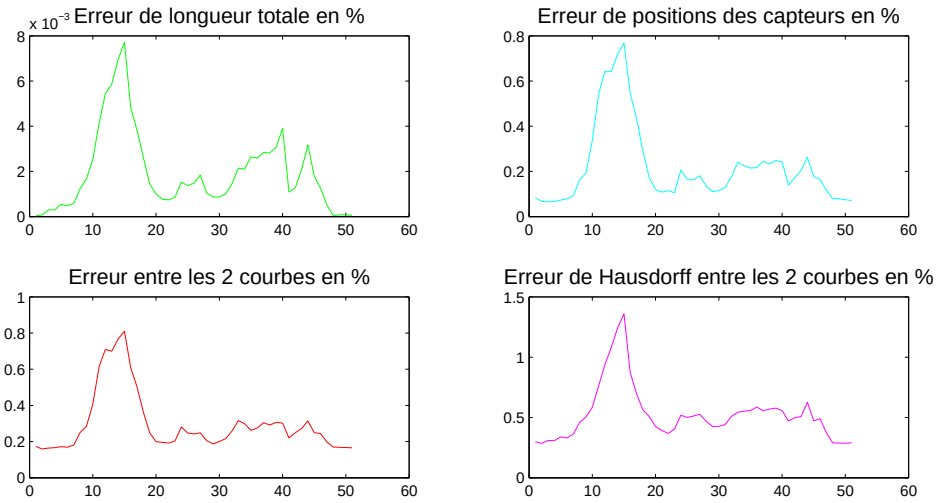
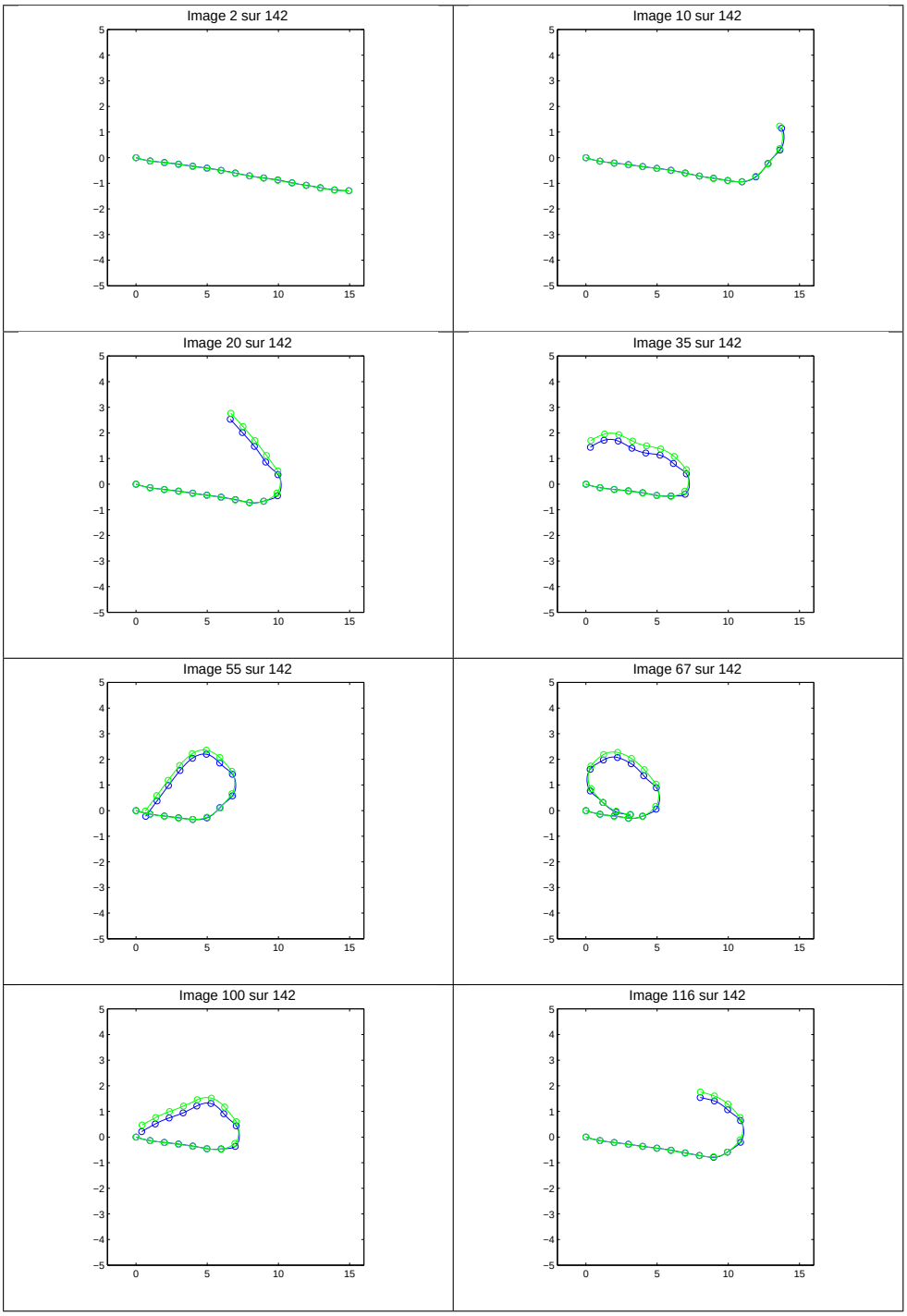


FIG. 5.14 – Erreurs du test 1 de déformation de courbe plane

Nous pouvons remarquer plusieurs faits à partir de ce test. Tout d'abord, les erreurs sont toujours très faibles donc nous pouvons considérer que les deux méthodes sont du même ordre de satisfaction. De plus, les erreurs sont les plus élevées lorsque la courbe a une forme assez oscillante (les erreurs maximales se situent pour l'image 15, où nous voyons la fin des deux courbes s'éloigner). Mais les courbes arrivent à retrouver une forme proche après s'être eloignées : nous n'avons pas de problème de divergence. Voyons les résultats obtenus pour un deuxième test.

Test 2 Le deuxième test montre une série de 142 images où nous enroulons le ruban sur lui-même et nous le déroulons ensuite. Voyons les résultats obtenus à certaines étapes de la déformation dans le tableau TAB.5.2



TAB. 5.2 – Test 2 de déformation de courbes planes avec MorphoSense

Nous voyons les deux courbes suivre le même mouvement de déformation. Nous calculons les erreurs entre les deux courbes à chaque pas de temps, les résultats sont représentés sur la figure FIG.5.15.

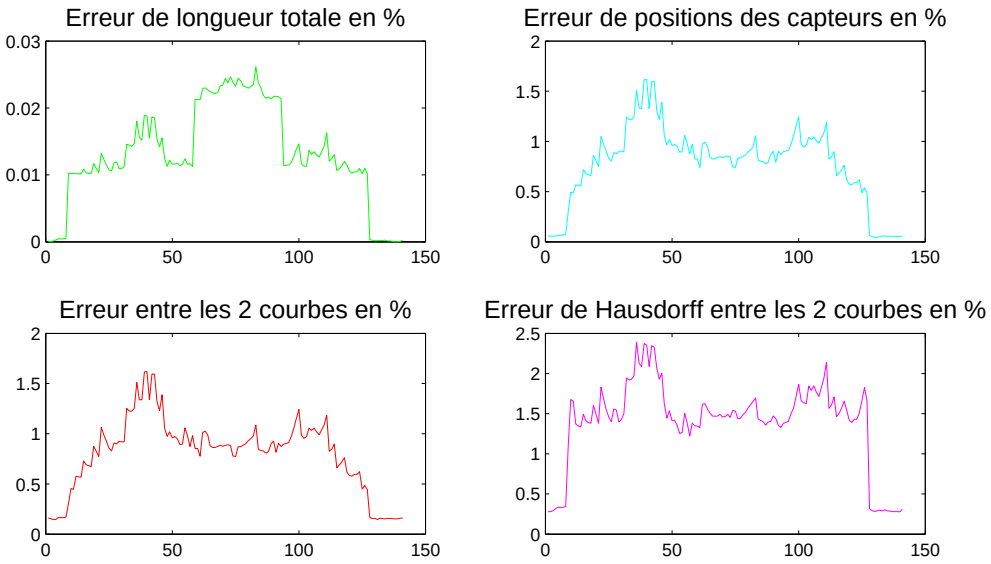
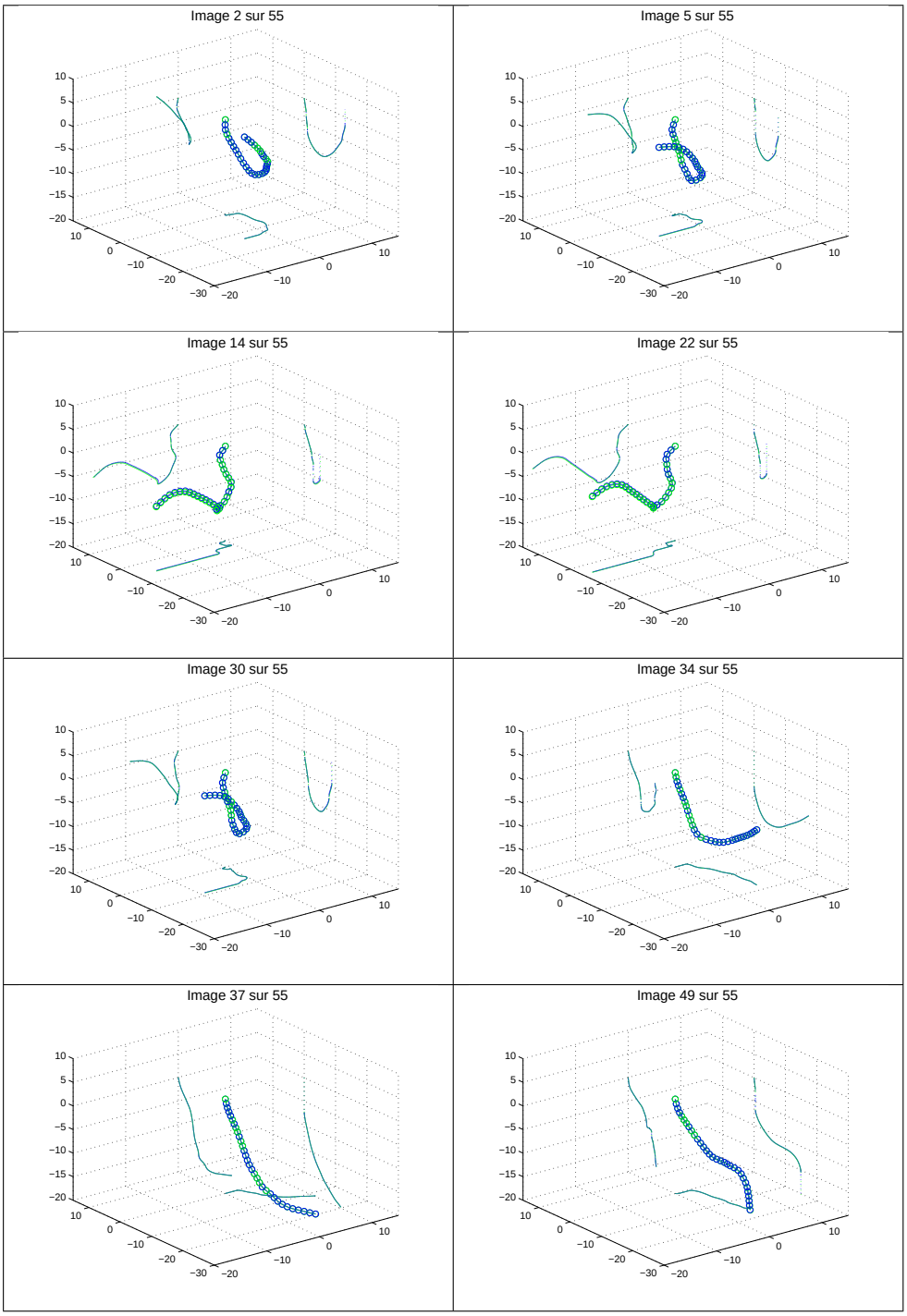


FIG. 5.15 – Erreurs du test 2 de déformation de courbe plane

Nous obtenons de faibles valeurs pour les quatre erreurs calculées, de l'ordre de $10^{-2}\%$ pour les longueurs, et entre 1 et 2% en ce qui concerne les erreurs de positions des courbes entre elles. Nous pouvons en conclure que les deux méthodes donnent de bonnes reconstructions des courbes planes en déformation. Voyons à présent ce que nous obtenons pour les courbes gauches.

Courbe gauche Nous déformons à présent le ruban dans l'espace. Nous obtenons une séquence de 55 images. Nous appliquons en parallèle la méthode de déformation et la méthode qui interpole la dérivée sur la sphère en chaque pas de temps. Nous obtenons ainsi deux courbes solutions à chaque étape que nous pouvons comparer. Voici quelques résultats visuels de courbes obtenues dans le tableau TAB.5.3. Sont tracées en bleu les courbes issues de la méthode d'interpolation sur la sphère et en vert celles obtenues par la méthode de déformation (nous traçons également les projections sur les trois plans parallèles pour mieux apprécier la forme spatiale).



TAB. 5.3 – Test de déformation de courbes gauches avec MorphoSense

Nous comparons les courbes obtenues en calculant les erreurs de longueur, de positions des capteurs, ainsi que l'erreur globale de position et celle de Hausdorff (voir figure FIG.5.16).

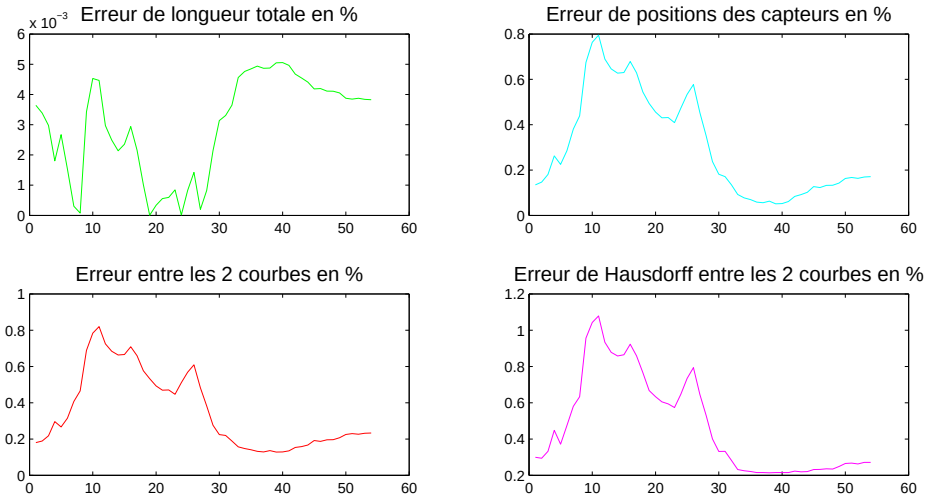
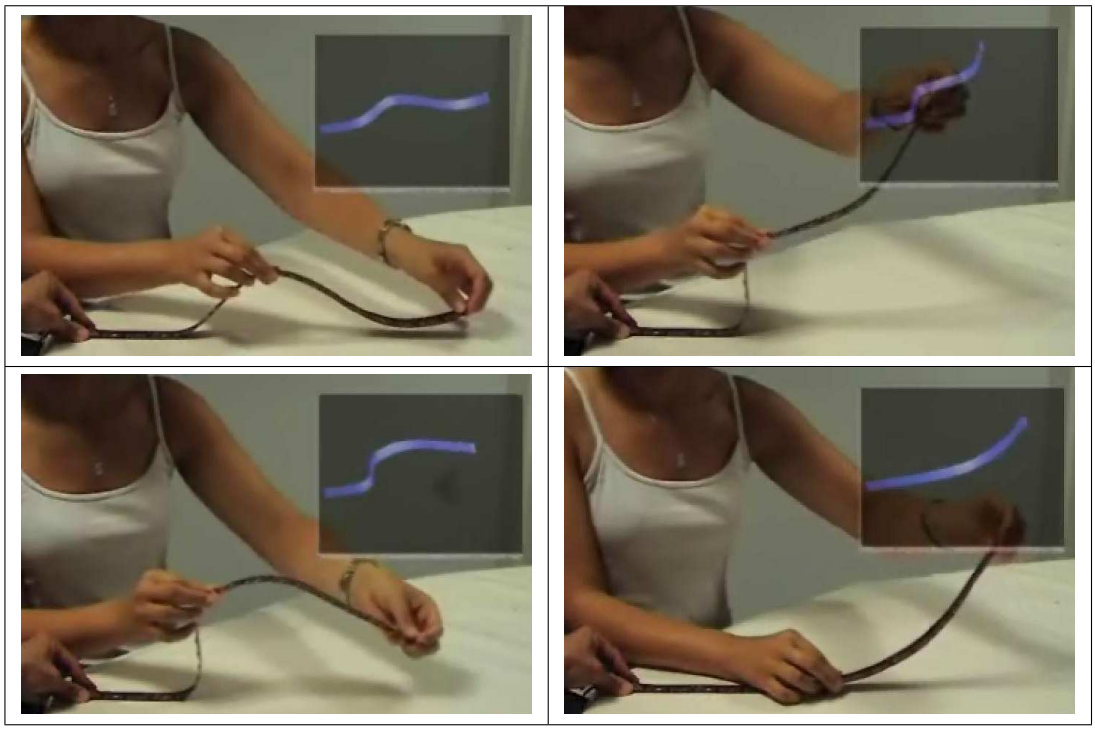


FIG. 5.16 – Erreurs du test de déformation de courbe gauche

Ces courbes d'erreur montrent de très bons résultats (erreurs inférieures au pourcentage) : les deux méthodes sont très satisfaisantes.

C. DÉMONSTRATEUR TEMPS RÉEL

Les algorithmes de reconstruction de courbes planes et courbes gauches ont été implémentés dans un environnement Java, afin d'utiliser le ruban MorphoSense en temps réel. Ainsi, nous pouvons faire bouger le ruban et voir un ruban virtuel suivre en temps réel les mêmes mouvements. Voyons quelques images illustrant ce logiciel et la reconstruction de courbes planes dans le tableau TAB.5.4 où nous voyons le ruban réel, et le ruban virtuel superposé.



Tab. 5.4 – Démonstrateur temps réel

Voyons à présent les résultats que nous obtenons pour la reconstruction de surface.

5.2.2 Les surfaces

Dans cette partie, nous allons juste montrer les premiers résultats que nous obtenons sur un exemple : nous aurons alors seulement des résultats visuels. Pour la reconstruction de surfaces, nous allons placer le ruban MorphoSense en différents endroits de la surface, dans la même direction, et nous appliquons alors notre algorithme de reconstruction de surfaces à partir de rubans instrumentés dans une seule direction.

Nous prenons comme surfaces tests certaines parties d'un coffre de toit.

Test 1

Nous essayons dans un premier temps de reconstruire la partie avant du coffre. Nous plaçons le ruban dans la direction vue sur la photo Fig.5.17.

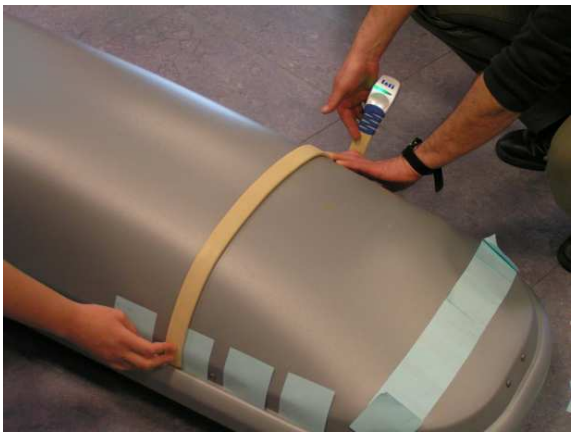


FIG. 5.17 – Devant du coffre de toit

Puis nous le décalons (aux positions notées sur la surface) afin d'obtenir 5 courbes. Nous utilisons également les données des 5 capteurs de départ de ces courbes afin de créer un sixième ruban virtuel dont nous connaissons la répartition des capteurs en mesurant les distances relatives et les données tangentielles en prenant les données normales de ces capteurs : nous créons alors également ce ruban dans la direction orthogonale et il nous permet de situer les 5 autres rubans les uns par rapport aux autres.

Afin de créer la surface, nous devons construire le réseau de courbes complet, nous les créons sour forme de Hermite comme expliqué dans la partie concernée. Puis nous définissons la surface solution par les patchs de Coons.

Les résultats visuels sont représentés sur la figure FIG.5.18, nous voyons à gauche les 5 courbes colorées données par les rubans et le reste du réseau (nous coupons les courbes des rubans à la partie qui nous intéresse), et nous voyons sur la droite la surface obtenue.

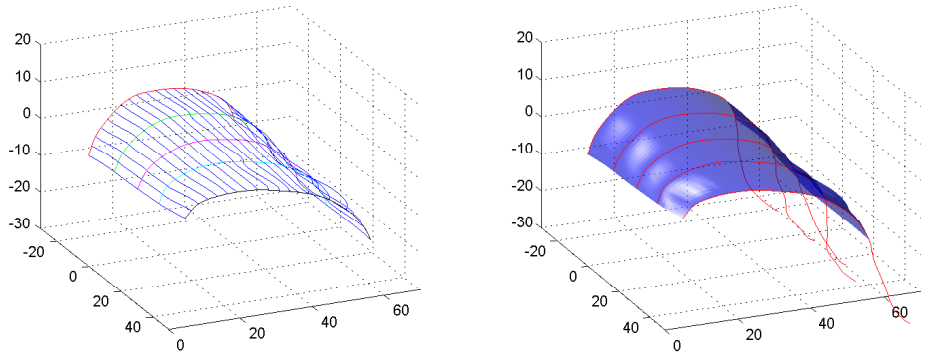


FIG. 5.18 – Réseau et surface reconstruite

Ces premiers résultats montrent la faisabilité de la méthode, à savoir la reconstruction de surfaces à partir de rubans instrumentés dans la même direction.

TEST 2

Pour le deuxième test, nous essayons de reconstruire la partie qui comporte de fortes variations de courbure (voir FIG.5.19).

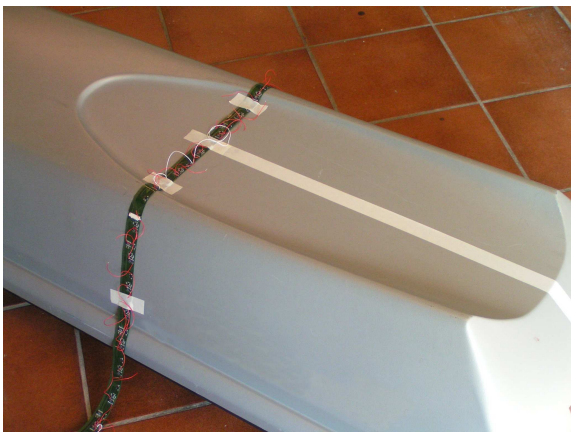


FIG. 5.19 – Arrière du coffre de toit

Nous reconstruisons ici le morceau de surface à partir de deux courbes placées dans la direction montrée sur la photo précédente. Les résultats sont visibles sur la figure FIG.5.20, où nous voyons un modèle numérique du coffre et la surface reconstruite à partir des deux rubans est superposée en marron.

FIG. 5.20 – Courbes et surface reconstruites à partir du prototype

Nous voyons dans cet exemple que les deux courbes noires montrent bien les variations de forme de la surface : il en découle que la surface reconstruite a une forme qui tient compte de ces variations.

5.3 Conclusion

Les avancées technologiques sont à l'origine du travail théorique de la thèse, et une réelle intéraction a été nécessaire pour mener à bien les différentes étapes dans ces deux domaines (nous voyons en particulier sur le problème des surfaces que les développements technologiques et théoriques sont indissociables).

Cette partie nous a permis de détailler le travail technologique réalisé en parallèle de cette thèse. Dans un premier temps, nous avons découvert les capteurs d'orientation réalisés au LETI, et nous avons déterminé la façon dont nous pouvions obtenir les données tangentielles dont nous avions besoin pour utiliser nos algorithmes de reconstruction. Un prototype a alors été réalisé à partir de ces conclusions.

Nos algorithmes ont ainsi pu être confrontés aux données réelles des capteurs. Un premier résultat positif a été de voir des reconstructions de courbes à partir du démonstrateur, et montrait ainsi la faisabilité d'une telle adéquation entre les algorithmes et ce système de capture de forme. Nous avons ensuite cherché à caractériser la qualité des courbes obtenues avec la création d'un gabarit, où nous pouvions ainsi comparer quantitativement la courbe plane à reconstruire et la courbe plane reconstruite : nous montrions alors des erreurs globales de position de moins de 2% en moyenne, ce qui permettait de valider nos méthodes sur ce premier prototype.

Nous avons ensuite testé nos différents algorithmes sur ce prototype, à savoir la reconstruction des courbes gauches (qui utilisait alors tous les capteurs du ruban), et les suivis de déformation de courbes planes et gauches, mais ceci exclusivement dans un aspect qualitatif, à savoir que nous n'avions pas la forme réelle du ruban à tout moment pour étudier réellement la qualité des reconstructions. Les algorithmes ont ensuite été portés en code Java, afin d'obtenir un démonstrateur temps réel de déformations de courbes.

Enfin, nous avons fait des premiers tests de reconstruction de surfaces à partir de ce ruban, en l'appliquant en différents endroits de la surface à reconstruire et en utilisant la méthode de reconstruction à partir d'une famille de rubans instrumentés dans la même direction. Les résultats montrent pour l'instant la faisabilité d'un tel processus de reconstruction.

Ces résultats montrent aussi la nécessité de toujours confronter la méthodologie d'utilisation des capteurs avec les potentialités théoriques et algorithmiques, afin de trouver les meilleurs choix de développements communs.

Un dernier point dont nous avons peu parlé concerne l'aspect des pré-traitements des données. Ils peuvent être de plusieurs ordres : trouver de meilleures fonctions de mélange pour obtenir les données tangentielles plus fiables, éliminer des données erronées, modifier les données des capteurs en tenant compte de leur précision, ou bien dans une autre direction tenir compte des perturbations des capteurs afin d'extraire les données tangentielles (par exemple dans le cas des accéléromètres, ils ne donnent leur propre orientation par rapport au champ gravitationnel que dans le cas où les mouvements appliqués au système sont

à accélération propre nulle : dans le cas où les mouvements seraient soumis à une accélération, nous devons alors extraire cette accélération pour obtenir l'information tangentielle recherchée). C'est un travail d'envergure, en marge de ce travail purement théorique, mais qui sera nécessaire si les technologies doivent être utilisées avec une grande fiabilité.

Conclusion et Perspectives

Cette thèse s'inscrit dans un contexte de capture de forme. Le LETI a développé des capteurs capables de s'orienter dans l'espace et a ainsi permis d'ouvrir des perspectives dans la capture de mouvement. De plus leur miniaturisation a permis d'envisager de les intégrer en grande quantité pour équiper des structures 1D (rubans ou fils instrumentés) ou 2D (coques, tissus, voiles, ...) ouvrant alors la voie à ce que l'on peut appeler la capture de forme. Les techniques de résolution nécessitaient de nouvelles méthodologies de raisonnement. Des solutions ont alors pu être mises en œuvre avec la vision géométrique que nous avons apportée à ce problème.

A notre connaissance, cette vision géométrique est peu utilisée pour la capture de mouvement en général (basée principalement sur des systèmes vidéo, donc utilisant des méthodes issues de la vision), et c'est aussi vrai en ce qui concerne l'utilisation des capteurs pour la capture de mouvement (qui se base plus sur du traitement du signal). Ce travail apporte alors des solutions concrètes à divers problèmes (reconstruction de courbes, de surfaces,...) prouvant ainsi la faisabilité de l'approche et validant alors l'utilité de poursuivre le travail technologique sur les capteurs pour la capture de forme, de mouvement...

Nous avons formalisé ces problèmes technologiques en problèmes mathématiques : la question fut alors de reconstruire des courbes et des surfaces à partir de données tangentielles en des points de répartition connue.

Nous nous sommes en premier lieu penché sur le problème de reconstruction de courbes, où la formalisation des données par l'abscisse curviligne a permis de déterminer une méthode de reconstruction, pour laquelle nous avons extrait des propriétés de consistance (invariance par rotation et par homothétie, minimisation de quantités intrinsèques à la courbe), ainsi que des propriétés de convergence et de robustesse, qui ont permis de donner une rigueur physique aux courbes solutions obtenues par cette méthode.

Nous avons ensuite formalisé le problème de reconstruction de surfaces, pour lequel le système de capture de forme s'est organisé en système filaire, mais avec deux visions différentes : soit sous forme de filet de capteurs, soit sous forme de famille de rubans instrumentés dans une même direction. Nous avons alors proposé des méthodes de reconstruction pour ces deux systèmes, qui adoptaient la même stratégie : nous reconstruisons d'abord un réseau de courbes répondant aux contraintes du réseau de capteurs, puis nous définissons la surface s'appuyant sur ces courbes caractéristiques. Nous avons testé ces différentes méthodes sur de nombreux exemples, cela nous a permis de voir les possibilités qu'apportait chaque reconstruction. Nous avons également pu juger du caractère convergent des

méthodes apportées en étudiant le comportement des erreurs en fonction du nombre de capteurs. Cette partie a clairement montré que l'aspect technologique, à savoir comment équiper une surface par des capteurs pour en permettre la reconstruction, était indissociable de la formalisation mathématique.

Une fois ces méthodologies de capture de forme appréhendées, nous avons étendu la problématique au suivi de déformation de courbes et de surfaces, perspective naturelle dans un tel domaine d'application. Nous avons alors mis au point de nouvelles méthodes de reconstruction, qui permettent d'utiliser les connaissances de position de la forme à l'instant précédent dans une optique d'optimisation de temps de calcul. Nous avons ainsi pu comparer ces méthodes de déformation aux méthodes dites statiques développées précédemment. Une autre étape de l'amélioration des temps de calcul serait cependant probablement obtenue par une intégration plus poussée des différents éléments logiciels.

Nous avons enfin confronté notre travail de modélisation de courbes et de surfaces aux données fournies par les capteurs. Un prototype a ainsi été élaboré pour étudier la compatibilité de nos travaux avec ces données réelles et les résultats obtenus avec ce premier prototype prouvent la validité de ces méthodes de résolution, et ouvrent ainsi de grandes perspectives dans de nombreux domaines. En effet, nos travaux ont montré l'intérêt de cette recherche amont dans ce domaine des objets communicants, et de nombreux champs applicatifs s'ouvrent dans des domaines très variés, notamment dans les domaines du textile, de la CAO, de la santé, du sport ou encore dans l'aérodynamique (nous avions détaillé quelques exemples concrets en introdution).

Ces nouveaux outils sont dans ces cas des outils de mesure précis, filaires ou surfaciques, donnant des informations instantanées de comportement, et pouvant s'allier aux simulations pour appréhender un phénomène.

Pour aboutir à de tels objectifs, plusieurs perspectives sont à étudier dans la continuité de ce travail. A court terme et concernant directement le travail effectué ici, nous allons nous pencher sur les optimisations des méthodes développées, notamment celles qui concernent les déformations de courbes et surfaces. Ceci peut dans un premier temps se concevoir seulement en étudiant les optimisations d'un aspect logiciel, mais également dans un deuxième temps chercher à améliorer nos méthodes de déformations de courbes et surfaces (ceci peut alors impliquer de nouvelles contraintes à appliquer à notre système ou de nouvelles méthodes de résolution...).

Un point à étudier concerne également tout ce qui est en amont de nos méthodes de reconstruction de formes à partir des capteurs. Cela touche aussi bien le positionnement des capteurs (comment traiter au mieux les informations des capteurs s'ils ne sont pas en des points confondus), que la gestion des erreurs de précision des données issues des capteurs, ou le traitement des capteurs erronés.... Mais cela touche aussi les problématiques de suivi de déformations en vraie dynamique cette fois-ci. Nous avons vu que les capteurs que nous utilisons sont alors perturbés par l'accélération propre du mouvement. Une direction envisagée pour résoudre ce problème est alors de chercher un prétraitement des données pour extraire l'information tangentielle à l'information perturbée des capteurs, mais une autre possiblité pourrait être de chercher de nouvelles méthodes prenant en compte ces accélérations-là et fournissant alors des déformations absolues dans l'espace.

A moyen et long terme se posent des questions liées aux avancées technologiques, qui guident alors les perspectives mathématiques, et réciproquement. Nous devons dans ce cas créer de nouveaux prototypes de capture de forme (notamment utiliser plusieurs rubans en parallèle et les fixer sur une surface, ou réfléchir à la mise au point d'un filet de capteurs...) qui permettront alors de déterminer les évolutions à apporter aux méthodes développées. De plus, nous pouvons également élargir l'application de nos méthodes par la mise au point et l'utilisation de nouveaux capteurs, par exemple nous pourrions décrire des déformations de courbes et de surfaces élastiques si nous avions des informations sur les élongations des matériaux utilisés. En outre, nous pouvons aussi être amenés à chercher des méthodes complètement nouvelles si la façon de répartir les capteurs n'est plus sous forme de maillage rectangulaire, notamment si nous déversons de façon dense mais aléatoire des capteurs sur la surface à reconstruire...

En conclusion, je dirais que cette thèse a été le premier pas d'une collaboration entre le monde mathématique et sa vision géométrique, et le monde technologique avec l'utilisation des capteurs, qui a porté ses fruits et ouvre de grandes perspectives pour continuer cette alliance.

Bibliographie

[ATK68] Kendall E. ATKINSON. On the order of convergence of natural cubic spline interpolation. *SIAM J. NUMER. ANAL.*, 5(1) :89–101, 1968.

[BAR77] R. BARNHILL. *Representations an approximation of surfaces in Mathematical Software III.* J.R. Rice Editor, Academic Press, 1977.

[BAR82] R. BARNHILL. Coons' patches. *Computers in Industry*, 3 :37–43, 1982.

[BAR91] Jaques BARANGER. *Analyse numérique.* Hermann, 1991.

[BOO78] Carl DE BOOR. *A practical guide to splines.* Springer Verlag, 1978.

[BS77] Carl DE BOOR and Blair SWARTZ. Piecewise monotone interpolation. *Journal of approximation theory*, 21 :411–416, 1977.

[CAR76] Manfredo P. DO CARMO. *Differential geometry of curves and surfaces.* Prentice Hall New Jersey, 1976.

[COO64] S. COONS. Surfaces for computer aided design, available as ad 663 504 from the national technical information service,springfield, va 22161. Technical report, M.I.T., 1964.

[COO74] S. COONS. *Surface patches and B-spline curves,Computer Aided Geometric Design,.* In R. Barnhill and R. Riesenfeld editors, Academic Press, 1974.

[DEH89] Randall L. DOUGHERTY, Alan EDELMAN, and James M. HYMAN. Nonnegativity-, monotonicity-, or convexity-preserving cubic and quintic hermite interpolation. *Mathematics of computation*, 52(186) :471–494, 1989.

[DEM96] Jean-Pierre DEMAILLY. *Analyse numérique et équations différentielles.* Presses Universtaires de Grenoble, 1996.

[FAR02a] Gerald FARIN. *Curves and Surfaces for CAGD, A practical Guide, fifth edition.* Academic Press, 2002.

[FAR02b] Rida T. FAROUKI. Pythagorean-hodograph curves. *Handbook of Computer Aided Geometric Design*, pages 405–427, 2002.

[FC85] F.N. FRITSCH and R.E. CARLSON. Monotone piecewise cubic interpolation. *SIAM J. NUMER. ANAL.*, 22(2) :386–400, 1985.

[GOR83] W. GORDON. An operator calculus for surface and volume modeling. *IEEE Computer Graphics and Applications*, 3 :18–22, 1983.

[HAL73] C. A. HALL. Natural cubic and bicubic spline interpolation. *SIAM Journal on Numerical Analysis*, 10(6) :1055–1060, 1973.

[HEN98] David W. HENDERSON. *Differential Geometry : A geometric Introduction.* Prentice Hall, 1998.

[HP04a] M. HOFER and H. POTTMANN. Energy-minimizing splines in manifolds. Technical Report 122, Geometry Preprint Series, Vienna Univ. of Technology, 2004.

[HP04b] M. HOFER and H. POTTMANN. Energy-minimizing splines in manifolds. *ACM Transactions on Graphics*, 23(3) :284–293, 2004.

[HYM83] James M. HYMAN. Accurate monotonicity preserving cubic interpolation. *SIAM J. SCI. STAT. COMPUT.*, 4(4) :645–654, 1983.

[JHA63] E. N. NILSON J. H. AHLBERG. Convergence properties of the spline fit. *Journal of the Society for Industrial and Applied Mathematics*, 11(1) :95–104, 1963.

[NIE04] Gregory M. NIELSON. ν-quaternion splines for the smooth interpolation of orientations. *IEEE transactions on visualization and computer graphics*, 10(2) :224–229, 2004.

[PH03] H. POTTMANN and M. HOFER. A variational approach to spline curves on surfaces. Technical Report 115, Institute of Geometry, October 2003.

[PH05] H. POTTMANN and M. HOFER. A variational approach to spline curves on surfaces. *Comput. Aided Geom. Design*, 22(7) :693–709, 2005.

[SHO85] Ken SHOEMAKE. Animating rotations with quaternions curves. *ACM Siggraph Computer Graphics*, 19(3) :245–254, 1985.

[WEI61] H. WEINBERGER. Optimal approximation for functions prescribed at equally spaced points. *J. Res. Nat. Bur. Standards*, 65(B) :99–104, 1961.

PUBLICATIONS

[Sprynski et al.(2005)] N. Sprynski, B. Lacolle, L. Biard, (2005), *Reconstruction de courbes à partir de données tangentielles*, Groupe de Travail sur la Modélisation Géométrique (GTMG), Poitiers, Mars 2005, pp. 61–75.

[Sprynski et al.(2006)] N. Sprynski, B. Lacolle, L. Biard, D. David, (2006), *Curves and Surfaces Reconstruction via Tangential Informations*, Sixth International Conference on Curve and Surface Design, Curve and Surface design, Avignon, July 2006, pp. 254–263.

[Sprynski et al.(2007)] N. Sprynski, B. Lacolle, L. Biard, D. David, (2007), *Curve Reconstruction via a Ribbon of Sensors*, Soumis à IEEE International Conference on Electronics, Circuits and Systems (ICECS), Marrakech, December 2007.

[Sprynski et al.(2007)] N. Sprynski, B. Lacolle, L. Biard, D. David, (2007), *Motion Capture of a Surface in deformation via sensors' array*, Article à soumettre,2007.

[Sprynski et al.(2007)] N. Sprynski, B. Lacolle, N. Szafran, L. Biard, (2007), *Surface Reconstruction via Geodesic Interpolation*, Article à soumettre,2007.

Annexes

Annexe A

Sur la reconstruction de la fonction d'angle

Sommaire

Nous nous intéressons dans cette partie à diverses reconstructions possibles de la fonction angle lors de la reconstruction de courbes planes à partir des données tangentielles. Pour cela, deux grandes options s'offrent à nous : l'interpolation des données ou leur approximation. C'est ce que nous allons détailler dans cette annexe.

A.1 INTERPOLATION "BASIQUE"

Nous allons ici nous intéresser à des interpolations polynomiales par morceaux des angles. La régularité de cette interpolation détermine la régularité de la courbe de reconstruction. En effet, si l'interpolation des angles est globalement C^k alors il en est de même pour la dérivée de la courbe, et la courbe solution est donc C^{k+1}. Ainsi, la méthode d'interpolation des angles va dépendre de la régularité que l'on veut contraindre à notre solution. Nous allons voir plusieurs possibilités de réponse : avec des interpolations linéaires par morceaux, quadratiques globalement C^1, quadratiques ayant seulement des raccords continus. L'interpolation par spline cubique ayant été choisie dans notre travail, elle est développée dans le corps de ce mémoire (voir section 1.1.2 en page 28).

A.1.1 INTERPOLATION LINÉAIRE PAR MORCEAUX

L'interpolation linéaire par morceaux est de la forme :

$$\alpha(s) = a_i \left(\frac{s - s_i}{h_i} \right) + b_i \tag{A.1}$$

pour s, tel que $s_i \leq s \leq s_{i+1}$, $i = 0, \ldots, n-1$. Nous avons ainsi $2n$ inconnues, et l'interpolant linéaire entre chaque point doit passer par les 2 points extrémités : le système est complètement déterminé

Les données sont déterminées par :

$$b_i = \alpha_i, i = 0, ..., n-1 \tag{A.2a}$$

$$a_i = \alpha_{i+1} - \alpha_i, i = 0, ..., n-1 \tag{A.2b}$$

A.1.2 INTERPOLATION QUADRATIQUE GLOBALEMENT C^1

Si nous souhaitons une solution finale C^2, alors nous allons interpoler la fonction des angles avec une spline de degré 2 par morceaux :

$$\alpha(s) = a_i \left(\frac{s - s_i}{h_i} \right)^2 + b_i \left(\frac{s - s_i}{h_i} \right) + c_i \tag{A.3}$$

pour s, tel que $s_i \leq s \leq s_{i+1}$, $i = 0, \ldots, n-1$. Nous avons ainsi $3n$ inconnues. Les contraintes sont :

- des contraintes de continuité ($\alpha(s_i^+) = \alpha_i, i = 0, \ldots, n-1$ et $\alpha(s_i^-) = \alpha_i, i = 1, \ldots, n$)

– des contraintes de continuité de la dérivée ($\alpha'(s_i^+) = \alpha'(s_i^-), i = 1, ..., n-1$).

Nous avons donc $3n$ inconnues et $3n-1$ contraintes; notre système a un degré de liberté : la dérivée au départ. Le système sera ainsi déterminé si l'on fixe le degré de liberté par minimisation de la courbure :

Les données sont alors complètement déterminées :

$$c_i = \alpha_i, i = 0, ..., n-1 \tag{A.4a}$$

$$\begin{cases} b_0 = \lambda \text{ :paramètre à déterminer} \\ a_0 = \alpha_1 - \alpha_0 - b_0 \end{cases} \tag{A.4b}$$

$$\begin{cases} b_i = \left(\frac{h_i}{h_{i-1}}\right)(2a_{i-1} + b_{i-1}) \\ a_i = \alpha_{i+1} - \alpha_i - b_i \end{cases} , i = 1, ..., n-1 \tag{A.4c}$$

Nous allons fixer le paramètre λ grâce à une contrainte globale : minimisation de la courbure $\int \alpha'^2$.Or :

$$\begin{aligned} \int_0^L \alpha'(s)^2 ds &= \sum_{i=0}^{n-1} \int_{s_i}^{s_{i+1}} \left(\frac{2}{h_i} a_i \left(\frac{s-s_i}{h_i}\right) + \frac{b_i}{h_i}\right)^2 ds \\ &= \frac{1}{3h} \sum_{i=0}^{n-1} a_i^2 + \frac{1}{h} \sum_{i=0}^{n-1} (a_i + b_i)^2 \\ &= \frac{1}{3h} \sum_{i=0}^{n-1} a_i^2 + CTE \end{aligned} \tag{A.5}$$

donc minimiser la courbure revient à minimiser la somme des carrés des a_i : $\sum_{i=0}^{n-1} a_i^2$. Calculons $\sum_{i=0}^{n-1} a_i^2$.

Trouvons une relation de récurrence pour la suite des a_i :

$$a_i = \alpha_{i+1} - \alpha_i - b_i \tag{A.6a}$$

$$b_i = \left(\frac{h_i}{h_{i-1}}\right)(2a_{i-1} + b_{i-1}) = \left(\frac{h_i}{h_{i-1}}\right)((\alpha_i - \alpha_{i-1}) + a_{i-1}) \tag{A.6b}$$

Nous obtenons :

$$a_i = \left((\alpha_{i+1} - \alpha_i) - \left(\frac{h_i}{h_{i-1}}\right)(\alpha_i - \alpha_{i-1})\right) - \left(\frac{h_i}{h_{i-1}}\right) a_{i-1} \tag{A.7}$$

En combinant ces différentes relations, nous pouvons exprimer a_i en fonction de λ.

$$\begin{aligned} a_i &= \sum_{k=1}^{i} \left((-1)^{i-k} \left(\frac{h_i}{h_k}\right) \left((\alpha_{k+1} - \alpha_k) - \left(\frac{h_k}{h_{k-1}}\right)(\alpha_k - \alpha_{k-1})\right)\right) \\ &\quad + (-1)^i \left(\frac{h_i}{h_0}\right)(\alpha_1 - \alpha_0) - \left(\frac{h_i}{h_0}\right)(-1)^i \lambda \end{aligned} \tag{A.8}$$

Chaque a_i est une fonction affine du paramètre, donc nous devons minimiser une fonction quadratique en λ.

$\sum_{i=0}^{n-1} a_i^2 = A\lambda^2 + B\lambda + C$, avec :

$$
\begin{aligned}
A &= \sum_{i=0}^{n-1} \left(\frac{h_i}{h_0}\right)^2 = \frac{1}{h_0^2}\sum_{i=0}^{n-1} h_i^2 \\
B &= \sum_{i=0}^{n-1} \left(\begin{array}{c} -2(-1)^i \left(\sum_{k=1}^{i} \left[(-1)^{i-k}\left(\frac{h_i}{h_k}\right)\left((\alpha_{k+1}-\alpha_k) - \left(\frac{h_k}{h_{k-1}}\right)(\alpha_k - \alpha_{k-1})\right)\right] \right. \\ \left. +(-1)^i \left(\frac{h_i}{h_0}\right)(\alpha_1 - \alpha_0)\right) \end{array} \right) \\
C &= \sum_{i=0}^{n-1} \left(\begin{array}{c} \sum_{k=1}^{i} \left((-1)^{i-k}\left(\frac{h_i}{h_k}\right)\left((\alpha_{k+1}-\alpha_k) - \left(\frac{h_k}{h_{k-1}}\right)(\alpha_k - \alpha_{k-1})\right)\right) \\ +(-1)^i \left(\frac{h_i}{h_0}\right)(\alpha_1 - \alpha_0) \end{array} \right)^2
\end{aligned}
$$

Nous obtenons le minimum de cette fonction quadratique lorsque la dérivée s'annule, c'est à dire en $\lambda = -\frac{B}{2A}$.

$$
\lambda = -\frac{1}{2\frac{1}{h_0^2}\sum_{i=0}^{n-1} h_i^2} \sum_{i=0}^{n-1} \left(\begin{array}{c} -2(-1)^i \left(\sum_{k=1}^{i} \left[(-1)^{i-k}\left(\frac{h_i}{h_k}\right)\left((\alpha_{k+1}-\alpha_k) - \left(\frac{h_k}{h_{k-1}}\right)(\alpha_k - \alpha_{k-1})\right)\right] \right. \\ \left. +(-1)^i \left(\frac{h_i}{h_0}\right)(\alpha_1 - \alpha_0)\right) \end{array} \right) \tag{A.9}
$$

En pratique, l'interpolation de fonctions par des splines quadratiques C^1 pose des problèmes d'oscillations dans certains cas, notamment lors d'alternances de phases constantes et de fortes variations. Ceci s'illustre dans la figure (FIG. A.1).

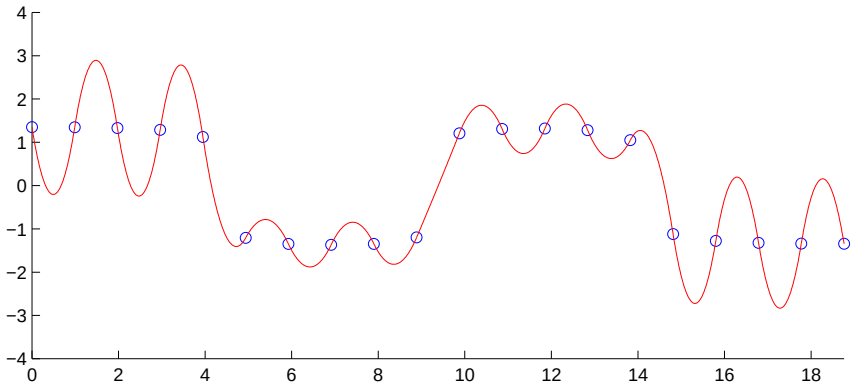


FIG. A.1 – Interpolation de degré 2 par morceaux à raccords C1

Il nous faut donc utiliser une autre interpolation. Par la suite, nous allons chercher une autre interpolation quadratique.

A.1.3 INTERPOLATION QUADRATIQUE SEULEMENT CONTINUE

La spline quadratique pose des problèmes d'oscillations lorsque nous forçons la continuité C^1. Nous allons donc chercher une autre interpolation quadratique, en cherchant d'autres contraintes possibles. Pour cela, nous pouvons nous baser sur les splines cubiques.

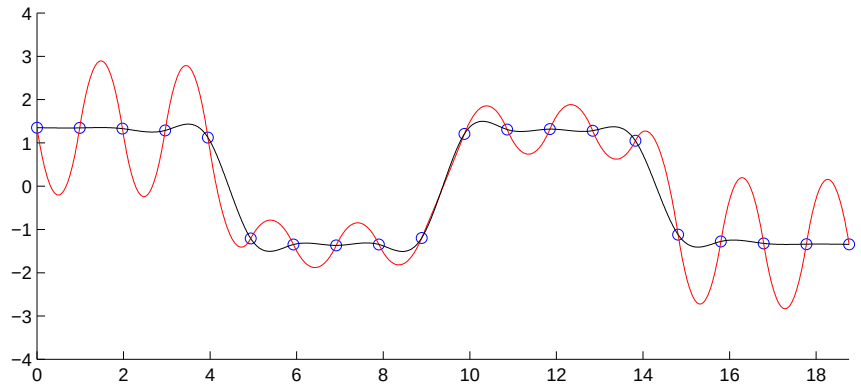


FIG. A.2 – Comparaison avec interpolation degré 3 par morceaux à raccords C2

En effet, celles-ci n'ont pas les problèmes d'oscillations vus précédemment (voir FIG. A.2).

La procédure que nous mettons en place ici est la suivante : nous allons interpoler les angles par une spline cubique de minimisation d'énergie, puis nous allons conserver les points milieux entre chaque intervalle, et enfin les morceaux de parabole de l'interpolation finale seront des morceaux de parabole passant par les deux points extrémités de l'intervalle et par le point milieu défini précédemment. Les contraintes sont les suivantes :

- continuité ($\alpha(s_i^+) = \alpha_i,\ i = 0, \ldots, n-1$ et $\alpha(s_i^-) = \alpha_i,\ i = 1, \ldots, n$)
- interpolation aux points milieux appartenant à la spline cubique d'interpolation ($\alpha\left(\frac{s_i+s_{i+1}}{2}\right) = \beta_i,\ i = 0, ..., n-1$)

Le système est complètement déterminé.
Les inconnues sont déterminées par :

$$c_i = \alpha_i,\ i = 0, ..., n-1 \tag{A.10a}$$
$$b_i = -\alpha_{i+1} + 4\beta_i - 3\alpha_i,\ i = 0, ..., n-1 \tag{A.10b}$$
$$a_i = 2(\alpha_{i+1} - 2\beta_i + \alpha_i),\ i = 0, ..., n-1 \tag{A.10c}$$

Voici un exemple d'une telle interpolation avec l'exemple précédent (FIG. A.3 : les points calculés avec la spline cubique sont indiqués).

Nous pouvons en quelque sorte dire que cette nouvelle interpolation est meilleure que la précédente, seulement nous perdons de la régularité (la courbe résultat est ainsi seulement C^2 par morceaux et à raccords C^1). Une dernière solution consiste à monter en degré : c'est ce que nous avons fait pour la mise en pratique de notre méthode en interpolant les angles avec une spline cubique.

A.1.4 CONCLUSION

Ces interpolations "basiques" ont toutes des caractéristiques différentes qui nous ont permis d'identifier les caractéristiques que nous voulons absolument avoir pour la reconstruction de la fonction angle : la méthode d'interpolation linéaire par morceaux a l'avantage

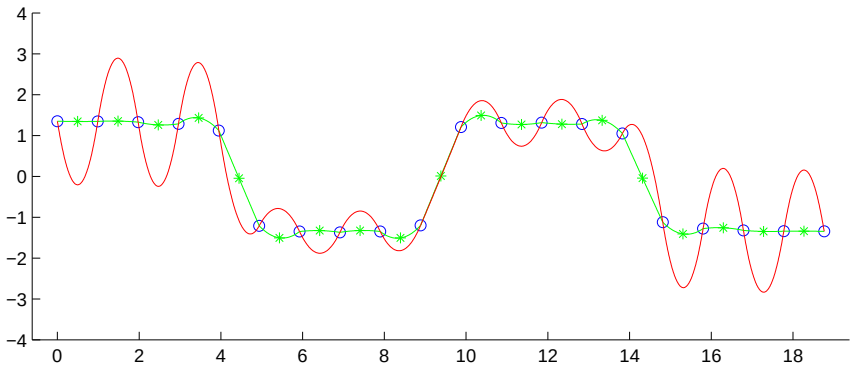


FIG. A.3 – Interpolation par degré 2 à raccords seulement C0

de garder la convexité de la courbe à reconstruire (mais elle s'éloigne de la courbe rapidement). La méthode d'interpolation par spline quadratique C^1 est satisfaisante pour les courbes de faible variation de courbure, mais diverge complètement dans les cas contraires. Dans ces cas-là, les deux autres méthodes donnent d'assez bons résultats, avec cependant toujours des erreurs moindres pour la spline quadratique C^0, cependant, elles font intervenir des maxima locaux dans la courbe (qui font donc intervenir des points d'inflexion supplémentaires dans la courbe à reconstruire, et modifie ainsi la forme globale de la courbe). Ceci nous permet de nous orienter vers des méthodes d'interpolation qui vont forcer la courbe à suivre la monotonie des données.

A.2 INTERPOLATION PAR CONTRAINTES DE FORMES

A.2.1 SPLINE CUBIQUE

Plusieurs articles écrits entre 1977 et 1989 posent le problème de conditions à respecter pour obtenir des courbes d'interpolation satisfaisant des contraintes de monotonie si les données possédent ces caractéristiques. Les splines cubiques étant complètement déterminées par les points d'interpolation et les tangentes en ces points, les conditions ont été exprimées sur les tangentes aux points d'interpolation. De Boor et Swartz dans [BS77] ont trouvé une condition nécessaire : ils obtiennent la "boite de de Boor-Swartz", dans laquelle les tangentes doivent se trouver, cela peut s'écrire ainsi :

$$0 \leq \lambda_i, \lambda_{i+1} \leq 3S_{i+1/2} \text{ ou } 3S_{i+1/2} \leq \lambda_i, \lambda_{i+1} \leq 0 \tag{A.11a}$$

$$\text{avec } \lambda_i = f'(t_i) \text{ et } S_{i+1/2} = \frac{f_{i+1} - f_i}{t_{i+1} - t_i} \tag{A.11b}$$

Puis, Fritsch et Carlson dans [FC85] ont élargi cette boite (la région acceptable est l'union de cette boite et d'une ellipse) dans laquelle la contrainte de conservation de monotonie est devenue une condition nécessaire et suffisante. Cependant, les algorithmes existant pour la reconstruction des splines se satisfont de la contrainte de De Boor-Swartz pour plus de

simplicité. Nous allons par la suite voir deux algorithmes différents (issus de [HYM83] et [DEH89]).

PREMIÈRE MÉTHODE

La première méthode est la méthode la plus "naïve" : elle consiste à calculer une spline cubique naturelle interpolant les points, et à modifier toutes les tangentes qui ne satisfont pas la contrainte de monotonie. La contrainte de monotonie pour λ_i est :

$$|\lambda_i| \leq 3\min(\left|S_{i-1/2}\right|, \left|S_{i+1/2}\right|) \tag{A.12}$$

Algorithme :

1. *Calculer la spline cubique naturelle, en déduire les tangentes en les points d'interpolation :* $\lambda_i, i = 0, ..., n$
2. *Boucler pour* $i = 0, ..., n$:
   $\lambda_i \leftarrow \begin{cases} \min(\max(0, \lambda_i), 3\min(\left|S_{i-1/2}\right|, \left|S_{i+1/2}\right|)) \text{ si } sgn(\lambda_i) \geq 0 \\ \max(\min(0, \lambda_i), -3\min(\left|S_{i-1/2}\right|, \left|S_{i+1/2}\right|)) \text{ si } sgn(\lambda_i) \leq 0 \end{cases}$
3. *Calculer la spline cubique associée aux tangentes ainsi définies.*

La figure suivante permet de voir la comparaison entre la spline cubique naturelle et la spline cubique qui conserve la contrainte de monotonie des données : (voir FIG.A.4)

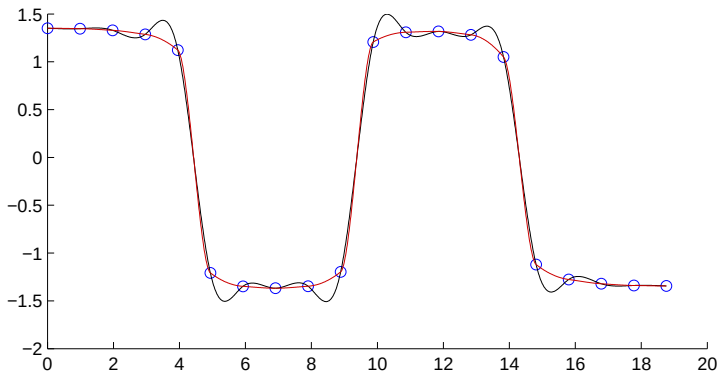


FIG. A.4 – Première méthode d'interpolation par contraintes de forme

Cette méthode a l'inconvénient de casser les raccords C^2 en tout point où l'on a modifié sa tangente. La méthode suivante essaie de minimiser le nombre de "pertes" de régularité.

DEUXIÈME MÉTHODE

Cette méthode se présente d'un point de vue plus global. Son algorithme est le suivant :

Algorithme :

1. *Calculer la spline cubique naturelle, en déduire les tangentes en les points d'interpolation :* $\lambda_i, i = 0, ..., n$

2. *Chercher le point i où la tangente est le plus loin de la contrainte (A.12) :*

$$|\lambda_i| - 3\min(|S_{i-1/2}|, |S_{i+1/2}|) = \max_j(|\lambda_j| - 3\min(|S_{j-1/2}|, |S_{j+1/2}|))$$

3. *Modifier cette tangente :*

$$\lambda_i \leftarrow \begin{cases} \min(\max(0, \lambda_i), 3\min(|S_{i-1/2}|, |S_{i+1/2}|)) \text{ si } sgn(\lambda_i) \geq 0 \\ \max(\min(0, \lambda_i), -3\min(|S_{i-1/2}|, |S_{i+1/2}|)) \text{ si } sgn(\lambda_i) \leq 0 \end{cases}$$

4. *Recalculer les deux splines cubiques entre $[t_0, t_i]$ et $[t_i, t_n]$ avec pour contraintes les tangentes aux bords.*

5. *Recommencer par récurrence sur chaque sous-intervalle.*

Cette méthode crée des coupures C^2 seulement aux endroits où l'on modifie les tangentes : la courbe obtenue est ainsi plus régulière qu'avec la méthode précédente.

La figure suivante permet de voir la comparaison entre la spline cubique naturelle et la spline cubique qui conserve la contrainte de monotonie des données : (voir FIG.A.5)

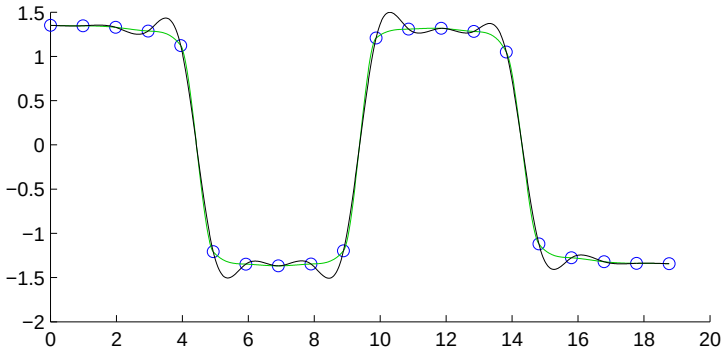


FIG. A.5 – Deuxième méthode

Nous avons pu remarquer que les splines ne sont pas satisfaisantes lorsqu'il y a de fortes variations de pente. Nous allons donc étudier d'autres splines qui permettent des variations de tangentes plus importantes que les splines polynomiales : les splines hyperboliques.

A.2.2 SPLINE HYPERBOLIQUE SOUS TENSION

Les splines hyperboliques sous tension permettent d'obtenir des variations de tangente plus importants que les fonctions polynomiales utilisées jusqu'à présent, et dans ([BAR91]) elles sont utilisées lorsque des contraintes de monotonie ou de convexité sont à respecter. Elles sont de la forme :

$$\alpha(s) = \alpha_{i+1}u_i + \alpha_i(1-u_i) + h_i^2\left(\beta_{i+1}\Psi_i(u_i) - \beta_i\Psi_i(1-u_i)\right) \tag{A.13}$$

pour s tel que $s_i \leq s \leq s_{i+1}, i = 0, ..., n-1$ et avec

$$h_i = s_{i+1} - s_i, u_i = \frac{s - s_i}{h_i}, \Psi_i(v) = \frac{\frac{sh(p_i v)}{sh(p_i)} - v}{p_i^2}, \text{ et } p_i = h_i r_i$$

Les paramètres à fixer sont les r_i, les inconnues sont les β_i, qui sont les dérivées secondes et qui sont déterminées de sorte que les morceaux se raccordent C^1.

Les inconnues à déterminer pour la construction de la spline hyperbolique sont les dérivées secondes, qui satisfont le système tridiagonal symétrique suivant :

$$A.B = D \tag{A.14}$$

avec

$$A = \begin{pmatrix} a_1 & b_1 & & & \\ b_1 & a_2 & b_2 & & \\ & b_2 & \ddots & \ddots & \\ & & \ddots & \ddots & b_{n-3} \\ & & & b_{n-3} & a_{n-2} \end{pmatrix}, B = \begin{pmatrix} \beta_2 \\ \beta_3 \\ \vdots \\ \beta_{n-2} \\ \beta_{n-1} \end{pmatrix}, D = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_{n-3} \\ d_{n-2} \end{pmatrix} \tag{A.15}$$

avec

$$a_i = h_i\Psi_i'(1) + h_{i+1}\Psi_{i+1}'(1), i = 1, ..., n-2 \tag{A.16a}$$

$$b_i = -h_{i+1}\Psi_{i+1}'(0), i = 1, ..., n-3 \tag{A.16b}$$

$$\Psi_i(v) = \frac{\frac{sh(p_i v)}{sh(p_i)} - v}{p_i^2}, i = 1, ..., n-1 \tag{A.16c}$$

$$d_i = \frac{y_{i+2} - y_{i+1}}{h_{i+1}} - \frac{y_{i+1} - y_i}{h_i}, i = 1, ..., n-2 \tag{A.16d}$$

et l'on ajoute les conditions $\beta_1 = \beta_n = 0$.

Cependant, cette méthode demande fixer des paramètres manuellement, et ne peut donc pas être appliquée de façon systématique.

D'autres méthodes sont envisageables pour la reconstruction d'une courbe d'après des points donnés : l'approximation.

A.3 APPROXIMATION PAR SPLINE DE LISSAGE

A.3.1 SPLINE DE LISSAGE : GÉNÉRALITÉS

Une spline de lissage est une spline cubique qui permet d'approcher les données $(s_i, \alpha_i), i = 0, ..., n$:

$$\alpha(s) = a_{1,i}\left(\frac{s - s_i}{h_i}\right)^3 + a_{2,i}\left(\frac{s - s_i}{h_i}\right)^2 + a_{3,i}\left(\frac{s - s_i}{h_i}\right) + a_{4,i} \tag{A.17}$$

avec s tel que $s_i \leq s \leq s_{i+1}, i = 0, ..., n-1$.(avec $h_i = s_{i+1} - s_i$)

Cette spline de lissage est déterminée en minimisant le critère :

$$J(\alpha) = \sum_{i=1}^{n} \frac{1}{\rho_i}(\alpha(s_i) - \alpha_i)^2 + \int_{s_1}^{s_n} (\alpha''(t))^2 dt \tag{A.18}$$

Les paramètres ρ_i jouent sur le coté approximation et lissage (si les $\rho_i \to 0$, nous obtenons une interpolation cubique des données, et à l'inverse, si les $\rho_i \to \infty$, nous obtenons la spline la plus lisse (correspondant au lissage au sens des moindres carrés)).

Si nous supposons les paramètres ρ_i déterminés, nous pouvons calculer cette spline de lissage voir [BAR91].

De manière succinte et pratique, nous devons résoudre le système pentadiagonal symétrique suivant :

$$A.U = D \tag{A.19}$$

avec

$$A = \begin{pmatrix} a_1 & b_1 & c_1 & & & & & \\ b_1 & a_2 & b_2 & c_2 & & & & \\ c_1 & b_2 & a_3 & b_3 & c_3 & & & \\ & \ddots & & \ddots & & \ddots & & \\ & & \ddots & & \ddots & & \ddots & \\ & & & \ddots & & \ddots & & c_{n-4} \\ & & & & \ddots & & \ddots & b_{n-3} \\ & & & & & c_{n-4} & b_{n-3} & a_{n-2} \end{pmatrix}, U = \begin{pmatrix} u_2 \\ u_3 \\ \vdots \\ u_{n-2} \\ u_{n-1} \end{pmatrix}, D = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_{n-3} \\ d_{n-2} \end{pmatrix} \tag{A.20}$$

avec :

$$a_i = \frac{h_i + h_{i+1}}{3} + \frac{\rho_i}{h_i^2} + \rho_{i+1}(\frac{1}{h_i} + \frac{1}{h_{i+1}})^2 + \frac{\rho_{i+2}}{h_{i+1}^2}, i = 1, ..., n-2 \tag{A.21a}$$

$$b_i = \frac{h_{i+1}}{6} - \frac{\rho_{i+1}}{h_{i+1}}(\frac{1}{h_i} + \frac{1}{h_{i+1}}) - \frac{\rho_{i+2}}{h_{i+1}}(\frac{1}{h_{i+1}} + \frac{1}{h_{i+2}}), i = 1, ..., n-3 \tag{A.21b}$$

$$c_i = \frac{\rho_{i+2}}{h_{i+1}h_{i+2}}, i = 1, ..., n-4 \tag{A.21c}$$

$$d_i = \frac{y_{i+2} - y_{i+1}}{h_{i+1}} - \frac{y_{i+1} - y_i}{h_i}, i = 1, ..., n-2 \tag{A.21d}$$

Ce système résolu nous permet de construire la spline de lissage :

$$\begin{aligned} f(t) = & \frac{(t_{i+1} - t)}{6}\left[\frac{(t_{i+1} - t)^2}{h_i} - h_i\right] u_i + \frac{(t - t_i)}{6}\left[\frac{(t - t_i)^2}{h_i} - h_i\right] u_{i+1} \\ + & \frac{(t_{i+1} - t)}{h_i} f(t_i) + \frac{(t - t_i)}{h_i} f(t_{i+1}), \text{ avec } t_i \le t \le t_{i+1}, i = 1, ..., n-1 \end{aligned} \tag{A.22}$$

et

$$f(t_i) = y_i - \rho_i\left(\frac{u_{i+1} - u_i}{h_i} - \frac{u_i - u_{i-1}}{h_{i-1}}\right), i = 1, ..., n-1, \text{ et } u_0 = u_1 = u_n = u_{n+1} = 0 \tag{A.23}$$

Nous devons à présent déterminer les paramètres ρ_i , afin d'obtenir une courbe qui soit proche des données, mais qui soit lisse au sens où elle ne fait pas intervenir les "artéfacts" qui apparaissent pour la spline cubique.

A.3.2 ALGORITHME PROPOSÉ

Le principe de cet algorithme est le suivant : nous voulons être très proche des données lorsqu'elles sont en forte variation, et être plus lisse dans les autres zones. Nous allons donc vouloir avoir des valeurs de paramètres très faibles lors de fortes variations des données.

Algorithme :

1. *Initialisation : on définit les* ρ_i *de départ comme suit :*

$$\rho_i = 10^{-2}.\frac{max - min}{|y_{i+1} - y_i|} \tag{A.24}$$

2. *On calcule la spline de lissage associée à ces paramètres : fonction* $\alpha(s)$.
   But : on va modifier ρ_i *si le point* i *est trop éloigné de la spline de lissage (trop éloigné d'après un paramètre* $VALEUR$ *déterminé).*
3. *Donc on boucle sur les* i *:*
   - *Calcul de l'*$ecart_i = |\alpha(s_i) - \alpha_i|$
   - *Si* $ecart_i > \frac{max-min}{VALEUR}$
     - *Taux de modification :* $taux = 10.\frac{ecart_i}{ecart_{\max}}$ *//le taux se situe entre 1 et 10.*
     - *Modification de* $\rho_i = \frac{\rho_i}{taux}$
4. *On obtient un nouveau vecteur de paramètres, on recalcule la spline et il se peut que certains points soient toujours trop éloignés de la courbe (car la modification d'un* ρ_i *rapproche le point i mais modifie les points autour) donc on boucle jusqu'à ce que les contraintes soient respectées*

Ainsi, le lissage des splines permet d'éliminer les artefacts qui apparaissent lors d'une interpolation cubique naturelle. L'algorithme élaboré permet de ne pas trop lisser afin de garder tout de même les endroits de la courbe où elle est soumise à de fortes variations.

Annexe B

Calcul de courbure et torsion

Cette annexe détaille les calculs faits pour exprimer la courbure et la torsion d'une courbe définie par l'abscisse curviligne. Sa fonction dérivée étant composée de vecteurs unitaires, nous pouvons l'exprimer en fonction des deux angles de la paramétrisation sphérique α et θ.

Pour ces calculs, nous utilisons les formules de Frenet :

$$\begin{aligned} C'(s) &= t(s) \\ C''(s) &= \kappa.n(s) \\ b(s) &= t(s) \wedge n(s) \\ b'(s) &= \tau.n(s) \end{aligned}$$

Nous allons donc calculer la courbure en calculant la norme des vecteurs $C''(s)$.

$$C''(s) = \begin{pmatrix} \alpha'(s)\cos(\alpha(s))\cos(\theta(s)) - \theta'(s)\sin(\alpha(s))\sin(\theta(s)) \\ \alpha'(s)\cos(\alpha(s))\sin(\theta(s)) + \theta'(s)\sin(\alpha(s))\cos(\theta(s)) \\ -\alpha'(s)\sin(\alpha(s)) \end{pmatrix}$$

En calculant la norme, nous obtenons :

$$\kappa = \sqrt{\alpha'^2 + \theta'^2 \sin^2 \alpha} \tag{B.1}$$

Pour calculer la torsion, il nous faut calculer le vecteur binormal ($b(s) = \frac{t(s) \wedge C''(s)}{\kappa(s)}$)), puis nous le dérivons, et pouvons ainsi calculer $\tau = -\langle n(s), b'(s) \rangle$.

$$\begin{aligned} b(s) &= \frac{1}{\kappa(s)} \begin{pmatrix} -\alpha'(s)\sin(\theta(s)) - \theta'(s)\sin(\alpha(s))\cos(\alpha(s))\sin(\theta(s)) \\ \alpha'(s)\cos(\theta(s)) - \theta'(s)\sin(\alpha(s))\cos(\alpha(s))\sin(\theta(s)) \\ \theta'(s)\sin(\alpha(s))^2 \end{pmatrix} = \frac{1}{\kappa(s)} V(s) \\ b'(s) &= \frac{-\kappa'(s)}{\kappa^2(s)} V(s) + \frac{1}{\kappa(s)} V'(s) \\ \tau &= -\left\langle n(s), \left(\frac{-\kappa'(s)}{\kappa^2(s)} V(s) + \frac{1}{\kappa(s)} V'(s) \right) \right\rangle . = 0 + \left\langle n(s), \frac{1}{\kappa(s)} V'(s) \right\rangle \end{aligned}$$

Et ainsi, après calculs, nous obtenons :

$$\tau = \frac{2\alpha'^2\theta' \cos\alpha + \alpha'\theta'' \sin\alpha - \theta' \sin\alpha \left(\alpha'' + \theta'^2 \cos\alpha \sin\alpha\right)}{\alpha'^2 + \theta'^2 \sin^2\alpha} \tag{B.2}$$

Annexe C

Simulations de courbes en mouvement

Sommaire

Cette partie utilise le travail développé dans [FAR02b].

C.1 CONSTRUCTION DE COURBES PLANES EN MOUVEMENT AVEC LES COURBES PH

C.1.1 MODÉLISATION D'UNE COURBE PH

Nous construisons une courbe PH cubique, pour cela, nous utilisons la formulation Bézier, avec donc 4 points de contrôle (voir FIG.C.1).

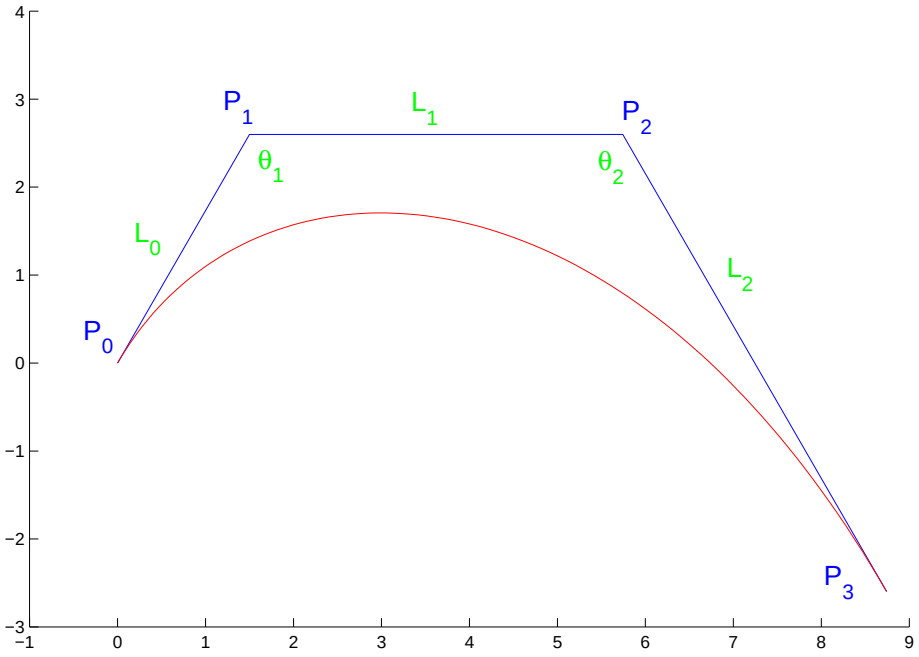


FIG. C.1 – Polygone de contrôle d'une cubique

Pour que ce soit une courbe PH, nous devons avoir

$$\begin{aligned} L_1^2 &= L_0 L_2 \\ \theta_1 &= \theta_2 = \theta \end{aligned}$$

Dans ce cas, nous avons la longueur de la courbe connue :

$$L = L_0 + L_2 - L_1 \cos(\theta)$$

Nous définissons dans ce cas une courbe cubique PH par 3 paramètres : L_0, L_2, θ.(aux rotations près).

Nous pouvons ainsi défnir les coordonnées des points de contrôle à partir de ces paramètres :

$$\begin{aligned} P_0 &= \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\ P_1 &= \begin{pmatrix} 0 \\ L_0 \end{pmatrix} \\ P_2 &= P_1 + L_1 \begin{pmatrix} \cos(\theta - \frac{\pi}{2}) \\ \sin(\theta - \frac{\pi}{2}) \end{pmatrix} \\ P_3 &= P_2 + L_2 \begin{pmatrix} \cos(2\theta - \frac{3\pi}{2}) \\ \sin(2\theta - \frac{3\pi}{2}) \end{pmatrix} \end{aligned}$$

C.1.2 DÉFORMATION D'UNE PH

Nous voulons maintenant créer une courbe à partir de celle-ci, proche et de même longueur. Nous allons procéder de 2 manières : modifier L_0, garder θ à sa valeur initiale et en déduire la nouvelle valeur de L_2, ou modifier θ, garder L_0 à sa valeur initiale et en déduire la nouvelle valeur de L_2,

MODIFICATION DE L_0

Si nous prenons $L_0' = L_0 + \delta$, alors nous cherchons L_2' tel que la longueur ne soit pas modifiée :

$$L_0 + L_2 - \sqrt{L_0 L_2}\cos(\theta) = L_0' + L_2' - \sqrt{L_0' L_2'}\cos(\theta)$$

d'où :

$$L_2' - \sqrt{L_2'}\left(\sqrt{L_0 + \delta}\cos(\theta)\right) + \left(\delta + \sqrt{L_0 L_2}\cos(\theta) - L_2\right) = 0$$

Nous obtenons $\sqrt{L_2'}$ racine de $x^2 - x\left(\sqrt{L_0 + \delta}\cos(\theta)\right) + \left(\delta + \sqrt{L_0 L_2}\cos(\theta) - L_2\right)$.
Nous en déduisons L_2 puis L_1, d'où la nouvelles courbe.

MODIFICATION DE θ

Si nous prenons maintenant $\theta' = \theta + \delta$, nous cherchons également la nouvelle valeur de L_2 qui conserve la longueur :

$$L_0 + L_2 - \sqrt{L_0 L_2}\cos(\theta) = L_0 + L_2' - \sqrt{L_0 L_2'}\cos(\theta')$$

d'où :

$$L_2' - \sqrt{L_2'}\left(\sqrt{L_0}\cos(\theta + \delta)\right) + \left(\sqrt{L_0 L_2}\cos(\theta) - L_2\right) = 0$$

Nous obtenons ici $\sqrt{L_2'}$ racine de $x^2 - x\left(\sqrt{L_0}\cos(\theta + \delta)\right) + \left(\sqrt{L_0 L_2}\cos(\theta) - L_2\right)$.
Nous pouvons en déduire L_2 puis L_1 d'où la nouvelles courbe.

C.1.3 PH PAR MORCEAUX

MODÉLISATION

Afin d'avoir des courbes plus complexes, nous pouvons créer des courbes PH par morceaux :

Si nous lions une premiere courbe de paramètres L_0, L_2, θ, (avec les points de controle P_0, P_1, P_2, P_3 comme définis avant), à une deuximème courbe de paramètres L_3, L_5, α, (avec les points de controle P_3, P_4, P_5, P_5), alors nous avons $L_4 = \sqrt{L_3 L_5}$ (voir FIG.C.2) et nous pouvons déterminer toutes les coordonnées des points de contrôle :

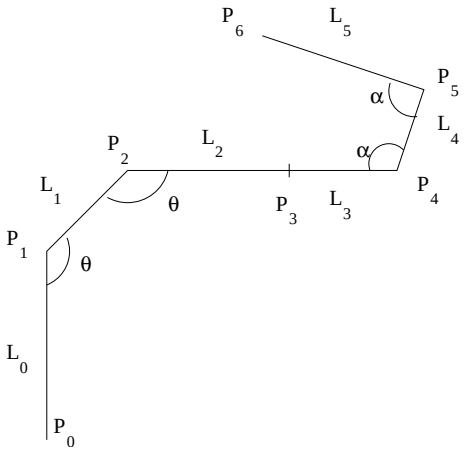


FIG. C.2 – PH par morceaux

$$
\begin{aligned}
P_4 &= P_3 + L_3 \begin{pmatrix} \cos(2\theta - \frac{3\pi}{2}) \\ \sin(2\theta - \frac{3\pi}{2}) \end{pmatrix} \\
P_5 &= P_4 + L_4 \begin{pmatrix} \cos(2\theta - \alpha - \frac{\pi}{2}) \\ \sin(2\theta - \alpha - \frac{\pi}{2}) \end{pmatrix} \\
P_6 &= P_5 + L_5 \begin{pmatrix} \cos(2\theta - 2\alpha + \frac{\pi}{2}) \\ \sin(2\theta - 2\alpha + \frac{\pi}{2}) \end{pmatrix}
\end{aligned}
$$

DÉFORMATION POUR LES TESTS

Pour les tests, nous avons appliqué une déformation avec une modification de l'angle pour le premier polygône de controle, et une déformation simultanée sur le deuxième polygone avec une modification de la longueur L_3.

Pour le test 1 Voici les paramètres utilisés pour le premier test. A l'étape initiale, nous avons $L_0 = 1$, $L_2 = 2$, et $\theta = \frac{2\pi}{3}$. Puis nous faisons varier θ tel que $\theta \leftarrow \theta + \delta$ où δ varie de 0.01 à 0.4 par pas de 0.01. Pour le deuxième polygone de controle, nous avons $L_3 = 6$, $L_5 = 5$, et $\alpha = \frac{3\pi}{7}$. Puis nous faisons varier L_3 tel que $L_3 \leftarrow L_3 + d$ où d varie de -0.1 à -4 par pas de -0.1.

Pour le test 2 A l'étape initiale, nous avons $L_0 = 1$, $L_2 = 5$, et $\theta = \frac{\pi}{2}$. Puis nous faisons varier θ tel que $\theta \leftarrow \theta + \delta$ où δ varie de 0.01 à 0.4 par pas de 0.01. Pour le deuxième polygone de controle, nous avons $L_3 = 2$, $L_5 = 3$, et $\alpha = \frac{4\pi}{5}$. Puis nous faisons varier L_3 tel que $L_3 \leftarrow L_3 + d$ où d varie de 0.1 à 4 par pas de 0.1.

Pour le test 3 A l'étape initiale, nous avons $L_0 = 5$, $L_2 = 6$, et $\theta = \frac{\pi}{3}$. Puis nous faisons varier θ tel que $\theta \leftarrow \theta + \delta$ où δ varie de 0.01 à 0.4 par pas de 0.01. Pour le deuxième polygone de controle, nous avons $L_3 = 2$, $L_5 = 5$, et $\alpha = \frac{3\pi}{5}$. Puis nous faisons varier L_3 tel que $L_3 \leftarrow L_3 + d$ où d varie de 0.1 à 4 par pas de 0.1.

Pour le test 4 En ce qui concerne le test 4, nous traçons une PH cubique en un seul morceau, avec les paramètres suivants : $L_0 = 3$, $L_2 = 9$, et $\theta = \frac{\pi}{2}$. Puis nous faisons varier θ tel que $\theta \leftarrow \theta + \delta$ où δ varie de -0.01 à -0.8 par pas de -0.05.

C.2 CONSTRUCTION DE COURBES GAUCHES EN MOUVEMENT AVEC LES COURBES PH

C.2.1 MODÉLISATION

Pour les courbes PH cubiques en 3 dimensions, nous avons le même style de contrainte que pour les courbes gauches. Pour un polygone de controle $P_0P_1P_2P_3$, avec pour longueurs de segments L_0, L_1 et L_2, et pour l'angle α entre les segments P_0P_1 et P_1P_2 et les segments P_1P_2 et P_2P_3, nous défnissons un autre angle ψ qui est l'angle entre $P_0P_1 \wedge P_1P_2$ et $P_1P_2 \wedge P_2P_3$, nous avons les contraintes suivantes :

$$\begin{aligned} \cos\psi &= \frac{2L_1^2}{L_0L_2} - 1 \\ L &= L_0 + L_2 - L_1\cos\alpha \end{aligned}$$

Nous pouvons alors définir une courbe PH dans un repère (x, y, z) qui satisfait les contraintes :

$$\begin{aligned} P_0 &= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \\ P_1 &= \begin{pmatrix} 0 \\ L_0 \\ 0 \end{pmatrix} \\ P_2 &= P_1 + L_1 \begin{pmatrix} \cos\left(\theta - \frac{\pi}{2}\right) \\ \sin\left(\theta - \frac{\pi}{2}\right) \\ 0 \end{pmatrix} \end{aligned}$$

et

$$P_3 = P_2 + \begin{pmatrix} x_3 \\ y_3 \\ z_3 \end{pmatrix}$$

tel que

$$\begin{aligned} x_3^2 + y_3^2 + z_3^2 &= L_2^2 \\ \langle P_2P_1, P_2P_3 \rangle &= L_1 L_2 \cos\alpha \\ \langle P_0P_1 \wedge P_1P_2, P_1P_2 \wedge P_2P_3 \rangle &= \|P_0P_1 \wedge P_1P_2\| \, \|P_1P_2 \wedge P_2P_3\| \left(\frac{2L_1^2}{L_0 L_2} - 1 \right) \end{aligned}$$

On choisit alors :

$$\begin{aligned} x_3 &= L_2 \sin\theta \cos\theta (-\cos\psi - 1) \\ y_3 &= x_3 \tan\theta + L_2 \\ z_3 &= \sqrt{L_2^2 - x_3^2 - y_3^2} \end{aligned}$$

C.2.2 Déformation

Comme pour les courbes planes, nous pouvons déformer en modifiant soit la valeur de L_0, soit la valeur de α. Nous pouvons supposer que nous gardons un angle ψ constant lors de la déformation pour avoir des formules simples (donc nous gardons une formule de L_1 en fonction de L_0 et L_2). Ce qui nous fait choisir $L_1 = \sqrt{kL_0L_2}$ avec k fixé. Nous obtenons ainsi des formules identiques à celles des courbes planes.

Modification de L_0

Si nous prenons $L_0' = L_0 + \delta$, alors nous cherchons L_2' tel que la longueur ne soit pas modifiée :

$$L_0 + L_2 - \sqrt{kL_0L_2}\cos(\theta) = L_0' + L_2' - \sqrt{kL_0'L_2'}\cos(\theta)$$

d'où :

$$L_2' - \sqrt{L_2'}\left(\sqrt{k\,(L_0+\delta)}\cos(\theta)\right) + \left(\delta + \sqrt{kL_0L_2}\cos(\theta) - L_2\right) = 0$$

Nous obtenons $\sqrt{L_2'}$ racine de $x^2 - x\left(\sqrt{k\,(L_0+\delta)}\cos(\theta)\right) + \left(\delta + \sqrt{kL_0L_2}\cos(\theta) - L_2\right)$.
D'où L_2 puis L_1, d'où la nouvelles courbe.

Modification de θ

Si nous prenons maintenant $\theta' = \theta + \delta$, nous cherchons également la nouvelle valeur de L_2 qui conserve la longueur :

$$L_0 + L_2 - \sqrt{kL_0L_2}\cos(\theta) = L_0 + L_2' - \sqrt{kL_0L_2'}\cos(\theta')$$

d'où :

$$L_2' - \sqrt{L_2'}\left(\sqrt{kL_0}\cos(\theta+\delta)\right) + \left(\sqrt{kL_0L_2}\cos(\theta) - L_2\right) = 0$$

Nous obtenons alors $\sqrt{L'_2}$ racine de $x^2 - x\left(\sqrt{kL_0}\cos(\theta+\delta)\right) + \left(\sqrt{kL_0L_2}\cos(\theta) - L_2\right)$. Nous en déduisons L_2 puis L_1 d'où la nouvelles courbe.

C.2.3 PH par morceaux

Modélisation

Afin d'avoir des courbes plus complexes, nous pouvons tout comme pour les courbes planescréer des courbes PH par morceaux. Pour cela, nous créons une autre courbe avec des caractéristiques L_3, L_4 et L_5 , des angles β et ϕ, et nous la déplaçons par rotation puis translation afin que elle se lie C1 avec la courbe précédente, (son polygone de controle est $P_3P_4P_5P_6$) tel que les points $P_1P_2P_3P_4$ soient co-planaires.

Déformation pour les tests

Pour les tests, nous avons appliqué une déformation avec une modification de l'angle pour la premier polygône de controle, et une déformation simultanée sur le deuxième polygone avec une modification de la longueur L_3.

Pour le test 1 Nous avons les paramètres suivants : $L_0 = 2$, $L_2 = 3$, $L_3 = 1$ et $L_5 = 5$, $\theta = \frac{3\pi}{5}$ et $\alpha = \frac{3\pi}{8}$. Nous réglons les paramètres pour les longueurs intermédiaires : $L_1 = \sqrt{\frac{3}{5}L_0L_2}$ et $L_4 = \sqrt{\frac{2}{5}L_3L_5}$. Puis nous faisons varier $\theta \leftarrow \theta + \delta$ avec δ de 0.05 à 0.5 par pas de 0.01 et $L_3 \leftarrow L_3 + d$ avec d de 0.2 à 0.2 par pas de 0.04.

Pour le test 2 Nous avons les paramètres suivants : $L_0 = 2$, $L_2 = 1$, $L_3 = 4$ et $L_5 = 5$, $\theta = \frac{3\pi}{4}$ et $\alpha = \frac{2\pi}{5}$. Nous réglons les paramètres pour les longueurs intermédiaires : $L_1 = \sqrt{\frac{1}{5}L_0L_2}$ et $L_4 = \sqrt{\frac{2}{5}L_3L_5}$. Puis nous faisons varier $\theta \leftarrow \theta + \delta$ avec δ de 0.07 à 0.7 par pas de 0.01 et $L_3 \leftarrow L_3 + d$ avec d de 0.5 à 0.5 par pas de 0.04.

Annexe D

Calibration de MorphoSense

Sommaire

Nous allons décrire dans cette partie comment nous pouvons obtenir les valeurs nécessaires à chaque capteur du ruban pour le calibrer.

Nous en premier lieu comment sont vraiment orientés les capteurs dont nous avons parlé précédemment. La figure FIG.D.1 montre l'orientation des accéléromètres, celle des magnétomètres et le repère lié au ruban dont nous cherchons à déterminer l'orientation.

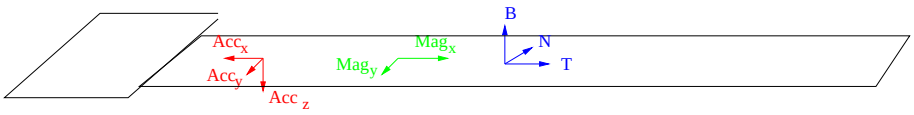


FIG. D.1 – Positionnement des axes sur le ruban

Pour la reconstruction des coordonnées du repère local (T, N, B) en chaque point de mesure, nous nous servirons alors des données $(-Acc_x, -Acc_y, -Acc_z, Mag_x, -Mag_y)$.

Pour la calibration de chaque capteur, nous n'avons en théorie besoin que des valeurs maximales (associées à 1) et minimales (associées à -1), mais nous allons également stocker les valeurs nulles (celles associées à 0), besoin qui se voit clairement aves les accéléromètres (voir le ruban horizontal lorsqu'il est placé sur une table).

Une fois que nous aurons ces 3 valeurs pour chaque capteur, nous pourrons déterminer sa valeur calibrée comprise entre -1 et 1 :

$$\text{si } val_{capt} < val_{nulle} \text{ alors} \qquad val_{calib} = \frac{val_{capt} - val_{nulle}}{val_{nulle} - val_{\min}}$$

$$\text{sinon} \qquad val_{calib} = \frac{val_{capt} - val_{nulle}}{val_{\max} - val_{nulle}}$$

D.1 CALIBRATION DES ACCÉLÉROMÈTRES TRI-AXES

Nous avons besoin ici de déterminer les 9 valeurs de calibration (3 valeurs maximales, 3 valeurs nulles et 3 valeurs minimales). Nous effectuons cela en 6 étapes.

D.1.1 VALEUR MIMIMALE DES Acc_z

Nous placons le ruban horizontal à plat à l'envers comme sur la figure FIG.D.2. Nous avons ainsi la valeur minimale de Acc_z, et les valeurs nulles de Acc_x et Acc_y.

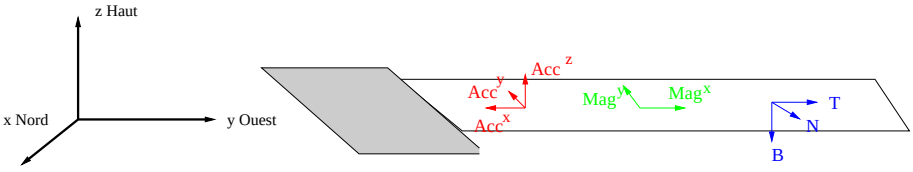


FIG. D.2 – Position minimale des az

D.1.2 VALEUR MINIMALE DES Acc_x

Nous placons le ruban vertical avec le boitier en haut, comme schématisé sur la figure FIG.D.3. Nous aurons ainsi la valeur minimale de Acc_x et la valeur nulle de Acc_z.

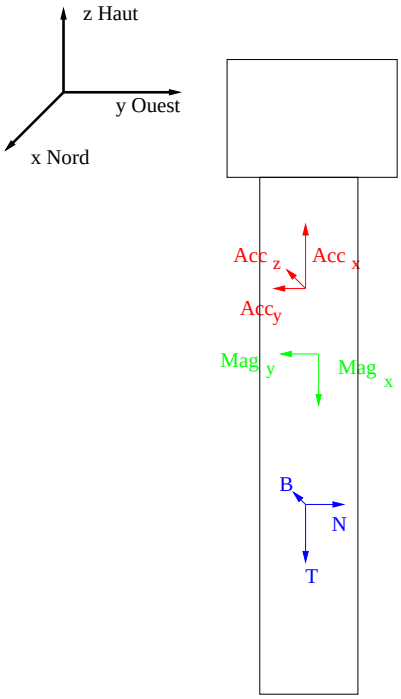


FIG. D.3 – Position minimale des ax

D.1.3 VALEUR MINIMALE DES Acc_y

Noua placons le ruban horizontal sur le coté avec le boitier à gauche de dos et nous aurons les valeurs minimales des Acc_y (voir FIG.D.4).

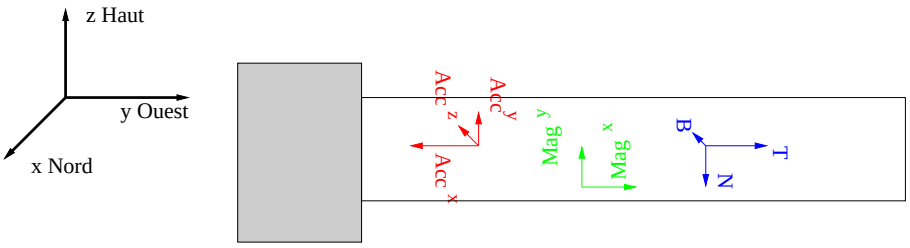


FIG. D.4 – Position minimale des ay

D.1.4 VALEUR MAXIMALE DES Acc_z

Pour avoir la valeur maximale des Acc_z, c'est la position opposée à la position minimale : horizontal à plat sur une table (voir FIG.D.5).

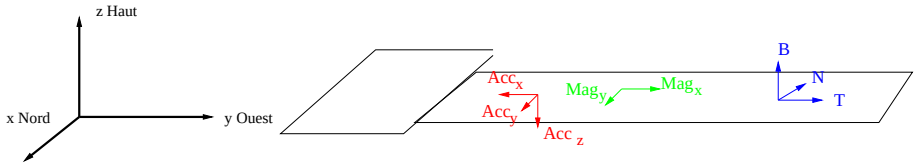


FIG. D.5 – Position maximale des az

D.1.5 VALEUR MAXIMALE DES Acc_x

Pour cette valeur, nous prenons également la position opposée à sa valeut minimlae : le ruban est vertical le boitier vers le bas (voir FIG.D.6).

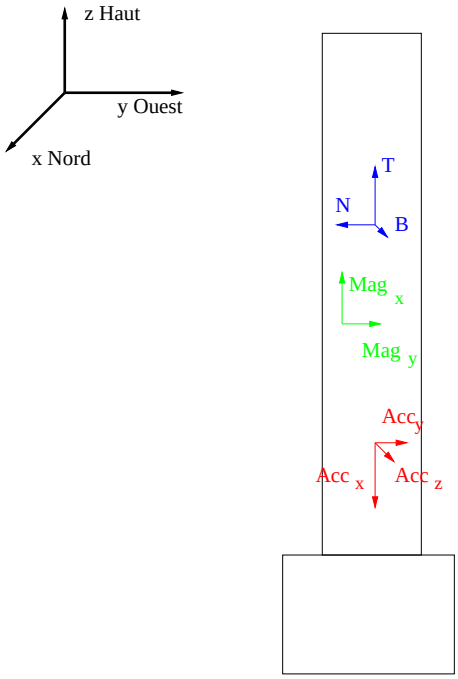


FIG. D.6 – Position maximale des ax

D.1.6 VALEUR MAXIMALE DES Acc_y

Enfin, le ruban est horizontal sur le coté avec le boitier à gauche vers nous (voir FIG.D.7).

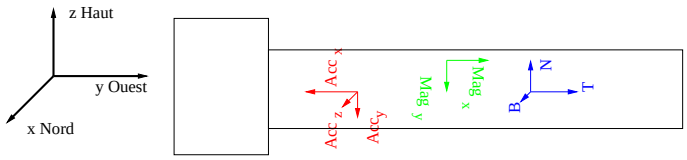


FIG. D.7 – Position maximale des ay

D.2 CALIBRATION DES MAGNÉTOMÈTRES BI-AXES

Pour les magnétomètres, nous devons déterminer 6 valeurs (2 valeurs maximales, 2 valeurs nulles et 2 valeurs minimales). Nous pouvons faire ceci en 4 étapes.

D.2.1 VALEUR MINIMALE POUR Mag_x

Nous devons orienter le capteur x dans le sens opposé au champ magnétique, (qui est dans le plan nord haut, de 60° vers le bas). Nous placons le ruban tout entier dans le sens du champ magnétique, avec le boiter en bas. Nous aurons ainsi la valeur minimale de Mag_x et la valeur nulle de Mag_y (voir FIG.D.8).

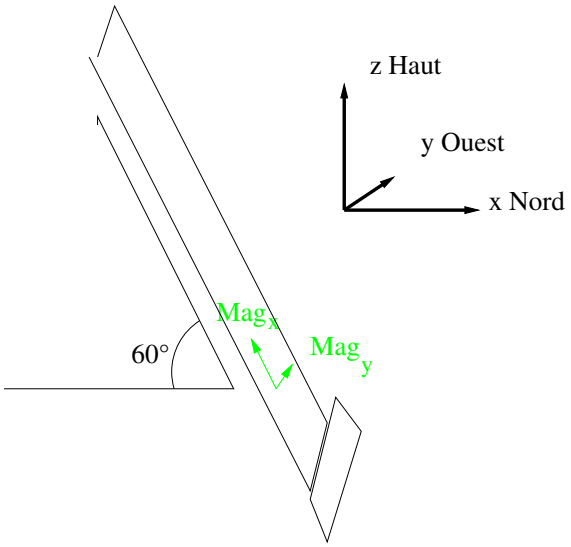


FIG. D.8 – Position minimale des magx

D.2.2 VALEUR MINIMALE POUR Mag_y

Nous le positionnons de sorte de magy soit minimale, donc en position orthogonale à la précédente (voir FIG.D.9).

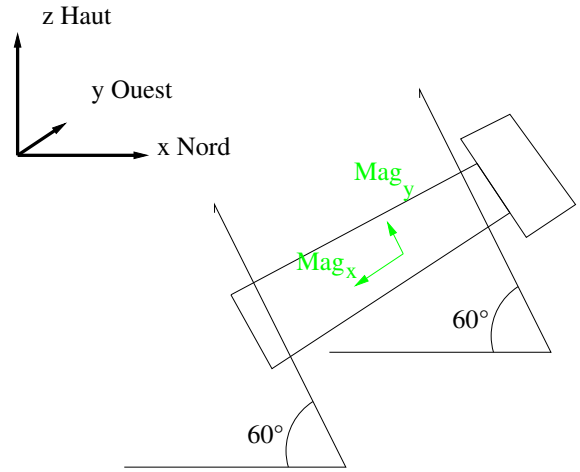


FIG. D.9 – Position minimale des magy

D.2.3 VALEUR MAXIMALE POUR Mag_x

Nous prenons la position boitier en haut pour avoir la valeur maximale (voir FIG. D.10)

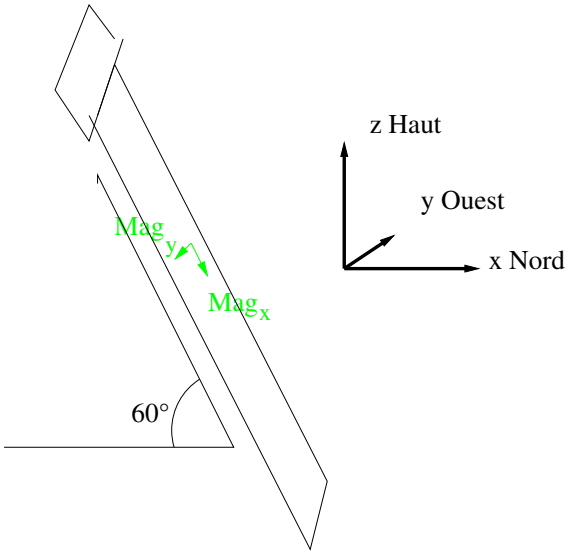


FIG. D.10 – Position maximale des magx

D.2.4 VALEUR MAXIMALE POUR Mag_y

Enfin, prenons la position opposée à celle pour la valeur minimale de Mag_y(voir FIG.D.11).

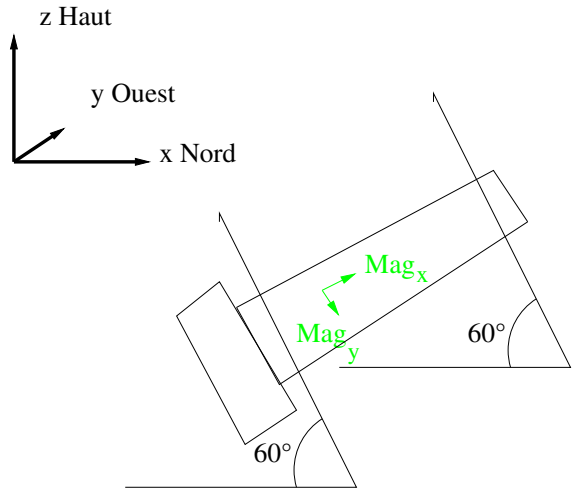


FIG. D.11 – Position maximale des magy

Nous venons de calibrer le ruban, il est donc opérationnel pour la reconstruction de courbes et de surfaces.