

# Étude d'un simulateur mécanique de tissus à partir de séquences d'images synthétiques

C. Syllebranque, S. Boivin

Projet ALCOVE, INRIA-Futurs Lille  
syllebra@lifl.fr, samuel.boivin@inria.fr

**Résumé :** De nombreux simulateurs permettent de reproduire des comportements physiques d'objets 3D. Certains peuvent être très complexes à paramétrer. Par exemple, il est très difficile, voire impossible, de retrouver les paramètres d'un simulateur de vêtements ou d'organes humains pour obtenir précisément un effet donné, et ce même pour un expert du domaine. C'est pourquoi il peut s'avérer judicieux de trouver ces paramètres directement à partir de séquences d'images réelles afin de reproduire en réalité virtuelle et de la façon la plus fidèle possible le comportement observé des objets du monde réel. Nous détaillons dans cet article une méthode d'estimation de paramètres à partir de séquences d'images. Nous réalisons également une étude de sensibilité de ces paramètres pour un simulateur mécanique de tissus à partir de vidéos synthétiques afin d'appliquer la méthode d'extraction aux séquences d'images réelles avant l'extension à d'autres domaines comme la simulation chirurgicale. Enfin, nous montrons quelques résultats obtenus par mécanique inverse sur deux scènes de tissus.

**Mots-clés :** analyse-synthèse d'images, identification, mécanique inverse, comparaison de vidéos, métrique d'erreur, minimisation d'erreur.

## 1 Introduction

### 1.1 Description du problème

De nombreux simulateurs permettent de reproduire des comportements physiques d'objets 3D, que ce soit en mécanique classique (animations de tissus), en mécanique des fluides (animation de feux, de fumées ou de liquides) ou encore en photométrie (visualisation photo-réaliste de scènes par exemple). Cependant, il est difficile, voire impossible, même pour un expert, de trouver les bons paramètres de ces simulateurs pour produire un effet donné, et tout particulièrement sur des objets réels complexes comme un oeil ou un foie par exemple. En effet, même si nous disposons d'un simulateur très puissant l'adéquation de ces paramètres avec le monde réel est très loin d'être évident, et dans le meilleur des cas il s'agit toujours de tâtonner pour obtenir le résultat voulu, et ce parfois pendant des heures. C'est pourquoi il pourrait être intéressant d'estimer directement ces paramètres depuis des séquences d'images réelles, afin de reproduire en réalité virtuelle et de la façon la plus fidèle possible le comportement observé des objets du monde réel. Nous détaillons dans cet article une méthode d'estimation de paramètres d'un modèle de tissus à partir de séquences d'images synthétiques.

### 1.2 Définition de la mécanique inverse

Le terme *mécanique inverse* est ici utilisé par analogie avec celui de *rendu inverse* [3]. Cette technique consiste à retrouver les propriétés photométriques d'objets réels (notamment leurs *fonctions de réflectance* [7][17]) à partir d'une ou plusieurs photographies et de la scène 3D correspondante. Dans le cas de la mécanique inverse, le principe est identique, à ceci près que la nature des paramètres à estimer est totalement différente. Il s'agira, par exemple, de retrouver les paramètres mécaniques<sup>1</sup> d'un oeil humain (rigidité, pression interne,...) à partir d'une séquence vidéo prise dans un bloc opératoire lors d'une intervention chirurgicale réelle.

## 2 État de l'art

### 2.1 Mécanique inverse

A ce jour, la seule tentative à notre connaissance présente dans la littérature pour mettre au point une méthode d'estimation de propriétés mécaniques d'objets réels à partir de vidéos est celle de Bhat et al.[2]. En effet, les

---

1. Nous pensons qu'il sera pour cela nécessaire de retrouver les paramètres photométriques au préalable en utilisant des méthodes de *rendu inverse* [3].

auteurs proposent une méthode pour retrouver les propriétés de divers tissus en s'appuyant sur une modification du modèle de Baraff et al.[1] conjugué au modèle de détection de collisions de Bridson et al.[5]. Le principe de l'estimation est ici semblable au rendu inverse. Il s'agit de générer une animation synthétique en utilisant des paramètres modifiés itérativement et de comparer, à chaque itération, la vidéo générée avec la vidéo initiale. Une description plus détaillée de cette méthode se trouve en section 3.1.

Dans la méthode de Bhat et al., la projection d'une lumière structurée sur le tissu lors de la capture de l'animation réelle permet, en appliquant une série de filtres (*steerable filters*, cf [8]), de retrouver l'orientation locale du tissu en chaque pixel. Il en résulte des cartes d'orientation locale (*angle maps*) dont la valeur est constante lorsque le tissu est plan et variable sur les plis et les creux du tissu. La comparaison de ces cartes constitue, avec une comparaison de la silhouette du tissu dans chaque animation, la métrique d'erreur (cf 3.2) utilisée par Bhat et al. pour comparer les deux vidéos (voir l'article original [2] pour plus de détails).

## 2.2 Estimation des paramètres par reconstruction 3D

Nous avons élaboré une nouvelle méthode d'estimation basée sur celle de Bhat et al.[2] décrite plus haut. Il s'agit d'effectuer le calcul de la différence entre deux animations au niveau du modèle géométrique (cf FIG 2). En effet, de nombreuses méthodes permettent de calculer la distance à l'aide d'invariants, c'est-à-dire une certaine quantification de la différence entre deux modèles 3D [19]. Cela permettrait en d'éviter l'opération de rendu qui peut être très coûteuse. Cependant, la donnée initiale est une séquence d'images. De la même manière que le rendu permet un passage de trois à deux dimensions, il est alors nécessaire d'effectuer l'opération inverse, c'est-à-dire retrouver la séquence de modèles géométriques 3D à partir de la vidéo initiale. Cette étape se nomme la *reconstruction 3D*, et de nombreux auteurs ont tenté de relever le défi.

L'un des problèmes majeurs lors de cette reconstruction 3D est l'effet des *inter-reflections diffuses*. Ce problème est maintenant bien connu dans le domaine de la synthèse d'images et il est reproduit avec de plus en plus de réalisme grâce aux algorithmes d'illumination globale, par exemple le *photon mapping* [14]. Cependant, dans le domaine de l'analyse d'images, retrouver la forme d'un objet à partir de son éclairage par exemple (*shape from shading* [12]), pose énormément de problèmes. Il est en effet très difficile de dissocier ces phénomènes de l'objet lui-même, car certaines informations proviennent d'autres objets et peuvent donc être hors champ.

Une autre méthode prometteuse consiste à déterminer des *shading primitives*, c'est-à-dire des motifs ou formes de bases prédéfinies. Une fois ces formes déterminées, une série représentant cette forme sous différentes conditions d'éclairage est générée en utilisant un modèle d'approximation des inter-reflections diffuses. Un réseau de neurones évolué est alors entraîné à l'aide de cette série. Il ne reste plus qu'à appliquer ce réseau sur les images pour détecter ces formes primitives. Les applications sont ici très variées : reconnaissance de formes, reconstitutions d'objets 3D, détections diverses. Actuellement, les plis (vêtements) et les cannelures sont deux primitives qui sont remarquablement bien détectées sur des images réelles [11][10]. Cependant, ces primitives sont très ciblées et, même dans le cas du tissu, il est impossible de détecter de manière robuste tous les plis générés, surtout lorsque ceux-ci sont de tailles très variables, et en particulier pour des animations.

Enfin, Torresani et al. [26] ont proposé très récemment une méthode permettant d'acquérir la forme d'un objet déformable à partir d'une vidéo. Nous avons tenté d'utiliser celle-ci dans notre cas sans succès. Elle semble prometteuse mais manque encore de robustesse dans des cas plus complexes comme la reconstruction d'une nappe tombant sur une table.

En prenant en compte ces considérations, nous nous sommes rendu compte qu'il est encore très difficile de réaliser cette reconstruction du modèle 3D et que les méthodes aujourd'hui proposées ne sont pas encore assez robustes pour les besoins de l'estimation de paramètres. Cependant, l'évolution de ces méthodes est à surveiller de près car cela permettrait un gain de performance très important lors de l'application de la mécanique inverse.

# 3 Notre méthode

## 3.1 Principe général

Le méthode est assez semblable à celle du rendu inverse[27][3]. Le but est de générer itérativement des séquences vidéos et de les comparer à la vidéo initiale (cf fig 1). Nous détaillons ici les données nécessaires et le fonctionnement de la méthode.

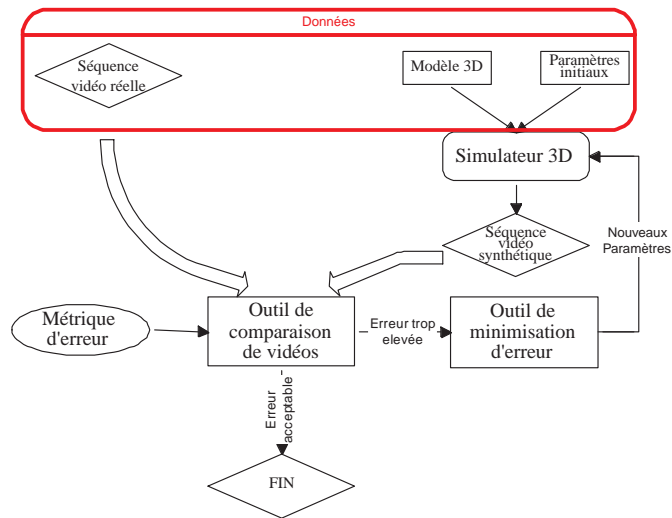


FIG. 1 – Vue schématisée du fonctionnement de la méthode d'estimation de propriétés physiques d'objets complexes depuis des séquences d'images.

### 3.1.1 Hypothèses et données

La mécanique inverse nécessite en entrée trois données. La première est la vidéo initiale à partir de laquelle l'estimation de paramètres sera faite. La seconde donnée est un modèle 3D de la scène vue dans cette séquence vidéo, à savoir tous les objets qui y sont visibles, les informations d'éclairage (positions et intensités des sources lumineuses) et les informations relatives à la caméra (position, focale, etc...). Les objets doivent également fournir leurs fonctions de réflectance [7][17] qui peuvent être retrouvées par rendu inverse [3], ou toute autre information permettant un rendu proche de la vidéo initiale comme, par exemple, la masse des objets ou encore les forces externes lorsque celles-ci ne peuvent pas être estimées directement à partir de la vidéo (par exemple lors d'une pression sur un organe à l'aide d'un outil chirurgical). Les valeurs initiales des paramètres recherchés constituent la dernière donnée. Elles peuvent être purement aléatoires, "semi-aléatoires", statistiques ou fixes selon le modèle d'objet complexe. Une fois toutes ces données définies, l'algorithme d'estimation de paramètres peut être utilisé.

### 3.1.2 Algorithme principal

Le principe de cette méthode est simple (fig 1). Il est décomposé en trois étapes qui seront répétées de manière itérative:

- Le *simulateur 3D* prend en charge la génération de la vidéo "synthétique" à l'aide des différentes données. Il peut s'agir d'un logiciel 3D polyvalent (*Blender*<sup>2</sup>, *3D Studio Max*<sup>3</sup> ou *Maya*<sup>4</sup> par exemple) ou encore d'une plateforme de simulation comme SPORE [20][21].
- Une fois l'animation synthétique générée, celle-ci est comparée à la vidéo initiale à l'aide d'une fonction reposant sur une métrique d'erreur (cf 3.2). Celle-ci doit être capable d'estimer une erreur permettant de quantifier la différence entre les deux animations données en entrée.
- L'erreur ainsi calculée est utilisée par un algorithme de minimisation d'erreur (cf 3.3). Ceci détermine un nouveau jeu de paramètres qui sera employé pour générer l'animation synthétique lors de l'itération suivante et permettra ensuite de minimiser la différence entre les deux séquences d'images.

Ce processus itératif se poursuit jusqu'à ce que l'erreur observée soit inférieure à un seuil prédéfini (ou jusqu'à un nombre maximal d'itérations défini au préalable par l'utilisateur). De cette manière, les paramètres estimés convergeront idéalement vers leurs valeurs optimales. Les résultats produits sont alors constitués d'une part des

<sup>2</sup>. *Blender* est logiciel de modélisation 3D et de rendu très répandu dans le monde des logiciels libres. Le site de la *freeBlenderfund* (<http://www.blender3d.com/>) donne tous les détails concernant ce logiciel.

<sup>3</sup>. *3D Studio Max* est un logiciel 3D polyvalent développé par *Discreet* (<http://www.discreet.com/>) intégrant, entre autres, des outils de modélisations, de texture et d'animation ainsi que divers modèles physiques.

<sup>4</sup>. *Maya* est la référence dans le domaine de l'infographie et plus généralement l'informatique graphique. Il est développé par *Alias* (<http://www.alias.com/>).

paramètres physiques estimés (obtenus à la dernière itération) et, d'autre part, d'une vidéo synthétique approximant au mieux la séquence originale (obtenue grâce aux paramètres physique précédemment calculés). Il est alors possible de comparer visuellement les deux animations et d'utiliser les paramètres trouvés pour les utiliser dans un autre contexte. Par exemple, les paramètres d'un modèle mécanique de tissu pourraient être estimés à partir d'une vidéo mettant en scène un simple morceau de l'étoffe, puis utilisés ensuite pour simuler un robe complète sur un personnage 3D.

### 3.2 Métrique d'erreur

La métrique d'erreur est fondamentale pour la méthode d'estimation. En effet, c'est elle qui va devoir quantifier la différence entre deux animations. Elle peut être comparée à notre œil qui peut nous dire si une vidéo "ressemble" plus ou moins à une autre, mais selon des critères mathématiques et statistiques bien définis, dans un souci de robustesse.

La comparaison d'images fait l'objet de nombreuses études, notamment dans le domaine de l'*indexation et recherche d'images* dans des bases de données vastes. Cependant, l'extension à la comparaison de séquences d'images reste encore très peu étudiée. Il semble très difficile de concevoir une métrique d'erreur générique pour comparer des vidéos, compte tenu de la nature très variée des types de simulateurs mécaniques possibles. En effet, les objets rigides et les fluides par exemple ne produisent pas du tout le même type d'images et leur structure est totalement différente. C'est pourquoi nous pensons que pour être efficace, la métrique doit être très spécialisée (détection de plis et cannelures pour le tissu par exemple [11]). Néanmoins, une approche basée sur une hypothèse des neurobiologistes stipulant que la perception visuelle humaine est sensible aux lignes et bordures en mouvement peut être utilisée pour concevoir une métrique plus générale. Cette approche est utilisée par Bhat et al. pour l'estimation de paramètres à partir de vidéo ([2], "A Metric for Matching Simulation to Video"). Nous détaillons brièvement cette métrique dans la section 2. C'est cette métrique que nous utilisons pour les expériences des sections suivantes. Nous pensons pouvoir l'appliquer à d'autres domaines comme la simulation de peau humaine.

Nous avons conçu une autre approche consisterait à faire dans un premier temps la reconstruction 3D à partir des vidéos [11][6] (cf 2.2) avant d'appliquer des algorithmes issus de l'identification de paramètres à partir des modèles géométriques [19][15]. Cependant, comme nous l'avons expliqué en section 2.2, les méthodes de reconstruction 3d actuelles ne sont pas encore assez robustes pour les besoins de la mécanique inverse.

### 3.3 Minimisation d'erreur

L'outil de minimisation d'erreur doit essayer de déterminer, à l'aide de l'erreur calculée par la métrique, les paramètres qui permettent à la vidéo synthétique (générée par le simulateur 3D avec ces nouveaux paramètres) d'approximer le plus possible l'animation initiale.

Il est ici possible d'utiliser de nombreuses méthodes qui peuvent dépendre du nombre de paramètres à minimiser. Lorsque le but est de déterminer un seul paramètre, on utilise des méthodes simples (*mono-objectif* [4]). Les méthodes *multi-objectifs*, comme le recuit simulé [16], peuvent quant à elles modifier plusieurs paramètres en même temps, ce qui nécessite une stratégie d'optimisation qui est propre à chaque application. Ces méthodes peuvent également être appliquées à un seul paramètre. Nous avons donc retenu ici le recuit simulé qui permet de balayer un large espace de recherche. Mais nous avons ici apporté une modification à la méthode de minimisation utilisée par Bhat et al.[2]. En effet, l'un de nos objectifs est d'étendre la mécanique inverse à des simulateurs beaucoup plus complexes, par exemple dans le domaine de la simulation chirurgicale. les modèles mécaniques utilisés peuvent alors nécessiter des dizaines de paramètres et il semble alors inconcevable, même en utilisant le recuit simulé, de balayer un espace de recherche aussi large pour trouver la solution optimale. Nous proposons donc ici de minimiser l'erreur générée par les paramètres un à un, dans un ordre défini par une hiérarchie de paramètres, qu'il convient d'établir après une étude de sensibilité des paramètres.

### 3.4 Étude de sensibilité des paramètres d'un modèle de tissu à partir de vidéos synthétiques

Après avoir réalisé une étude sur le comportement des paramètres d'un simulateur de tissu à paramétrisation simple (4 paramètres: résistance à la flexion, traction/compression, cisaillement et amortissement visqueux global, cf [25]), nous avons réalisé plusieurs expérimentations à partir de séquences d'images synthétiques. Une vidéo dite "initiale" est générée au préalable. Les paramètres du simulateur utilisés pour cette animation sont connus. Le but est alors de retrouver ces paramètres en utilisant la méthode d'estimation décrite sur le schéma FIG 1 avec une la marge d'erreur la plus réduite possible.

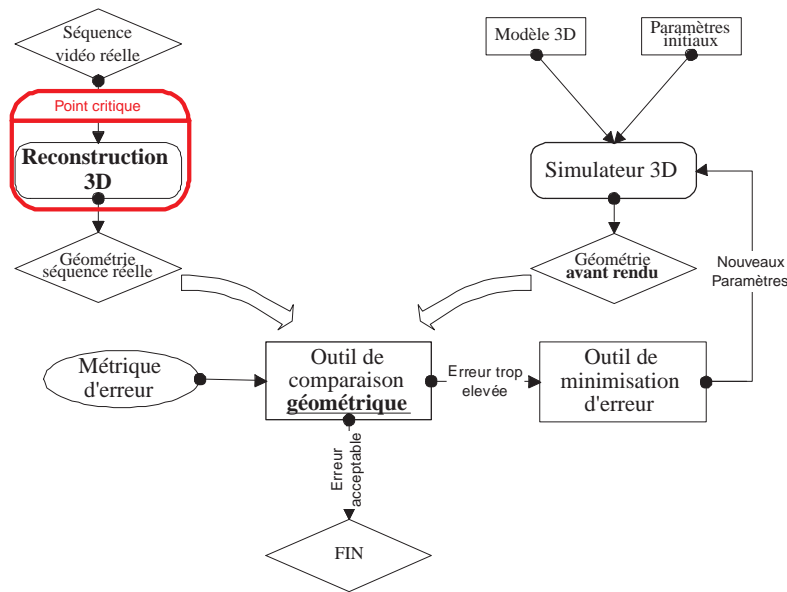


FIG. 2 – Vue schématisée du fonctionnement de la méthode d'estimation par reconstruction 3D : A la différence de la méthode illustrée FIG 1, la comparaison ne se fait pas au niveau des vidéos mais au niveau de la géométrie 3D. L'avantage ici est qu'il n'est plus nécessaire d'effectuer un rendu de l'animation pour **chaque itération**, mais l'opération inverse, à savoir la reconstruction 3d, **une et une seule fois**.

Tout d'abord, nous avons créé, pour chaque expérimentation, une fonction d'erreur de chacun des paramètres en balayant les espaces de validité<sup>5</sup>. Les courbes obtenues nous permettent alors d'affirmer, en vérifiant qu'elles ne comporte qu'un seul minimum global, que les paramètres peuvent être estimés séparément si ceux-ci sont décorrélés. Une illustration de la méthode qui utilisée pour obtenir ces courbes se trouve en FIG 3. Comme expliqué plus haut, nous utilisons la métrique d'erreur consistant à comparer les cartes d'orientation locales et la silhouette des tissus ainsi que le recuit simulé appliqué à la minimisation de l'erreur générée par les paramètres un à un.

Pour valider la minimisation d'erreur sur les paramètres successivement, il est nécessaire de montrer la décorrélation des paramètres entre eux. Le principe est le même que précédemment, à ceci près que deux paramètres sont variés à la fois. Une courbe d'erreur 3D est alors obtenue pour chaque couple de paramètre. Deux paramètres sont décorrélés s'ils n'influent pas l'un sur l'autre: la fonction d'erreur d'un paramètre conserve la même allure pour chaque valeur de l'autre paramètre.

Nous avons réalisé ces expérimentations à partir de deux scènes vidéos très différentes mettant en scène une nappe de tissu: un *drapé* (nappe tombant sur une table, fig 4a) et une sphère passant au travers d'une nappe suspendue par un bord (fig 4b). Lors de la génération de ces vidéos, nous simulons la projection d'une grille de lumière structurée afin de pouvoir réaliser le calcul des cartes d'orientation locale nécessaires à la métrique définie par Bhat et al. (cf section 2).

A l'aide de la méthode décrite plus haut, nous avons généré les fonctions d'erreur des paramètres pour chaque scène (fig 5). On voit sur ces courbes que chacune possède un minimum global. Cette observation nous permet d'affirmer que de telles fonctions sont "minimisables" sur les intervalles de validité, et donc qu'il est possible de retrouver une et une seule valeur pour chaque paramètre ayant été utilisé pour générer la vidéo initiale. Cependant, pour valider ce résultat mettant en jeu les paramètres séparément, il nous est nécessaire de prouver que les paramètres sont décorrélés entre eux.

Pour ce faire, nous avons testé les paramètres deux à deux de la même manière que précédemment, mais en faisant varier deux paramètres à la fois. Il en résulte 6 courbes en trois dimensions pour chaque expérience (fig 6 et 7). On peut noter à partir de ces surfaces que la variation d'un paramètre n'entraîne pas de changements radicaux sur l'allure de la fonction d'erreur de l'autre paramètre testé. En particulier, celle-ci conserve un minimum global sur l'intervalle testé. Nous en concluons que les paramètres du simulateur testé sont décorrélés entre eux pour les

5. Ici, les 3 paramètres non testés prennent leur valeur réelle

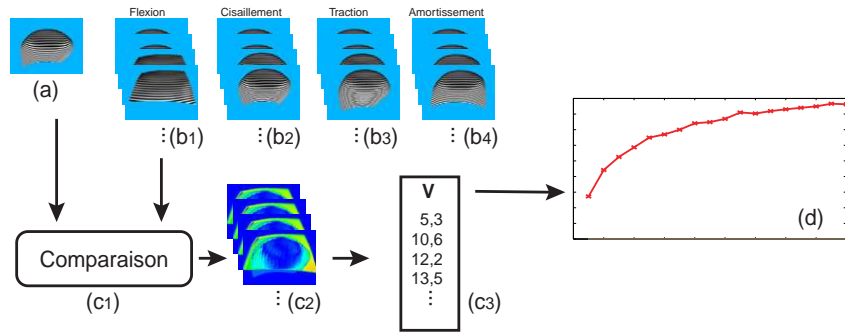


FIG. 3 – Principe de l'étude de sensibilité : (a) Une animation initiale avec les paramètres initiaux connus est tout d'abord générée. (b) Ensuite,  $n$  animations balayant l'espace de validité  $[b_{inf}; b_{sup}]$  de chaque paramètre (pas  $\Delta p = \frac{b_{sup} - b_{inf}}{n}$ ) sont également générées. (c) Celles-ci sont comparées avec l'animation initiale ( $c_1$ ) ce qui crée une série d'animations de différences ( $c_2$ ) et une liste des valeurs quantifiant ces différences ( $c_3$ ) d'où est extraite une courbe d'erreurs pour ce paramètre (d). Ce processus recommence ensuite (à partir de  $c_1$ ) pour les autres paramètres, ce qui peut être effectué grâce à la décorrélation des paramètres (fig 6 et 7).

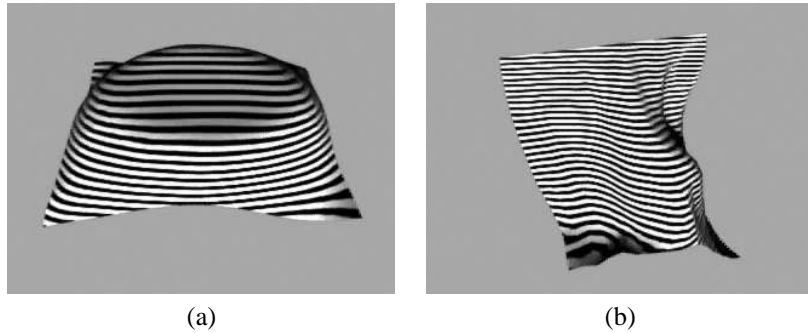


FIG. 4 – Capture des scènes utilisées pour les expérimentations: (a) Une nappe est maintenue au dessus d'une table puis lâchée sur celle-ci. Cette scène est connue sous le nom de drapé. (b) Une sphère est passée au travers d'une nappe suspendue par un bord. NOTE: On peut voir sur ces captures la lumière structurée en lignes projetée sur la nappe. Celle-ci est utilisée par la métrique de Bhat at al.[2] pour calculé les cartes d'orientation locale.

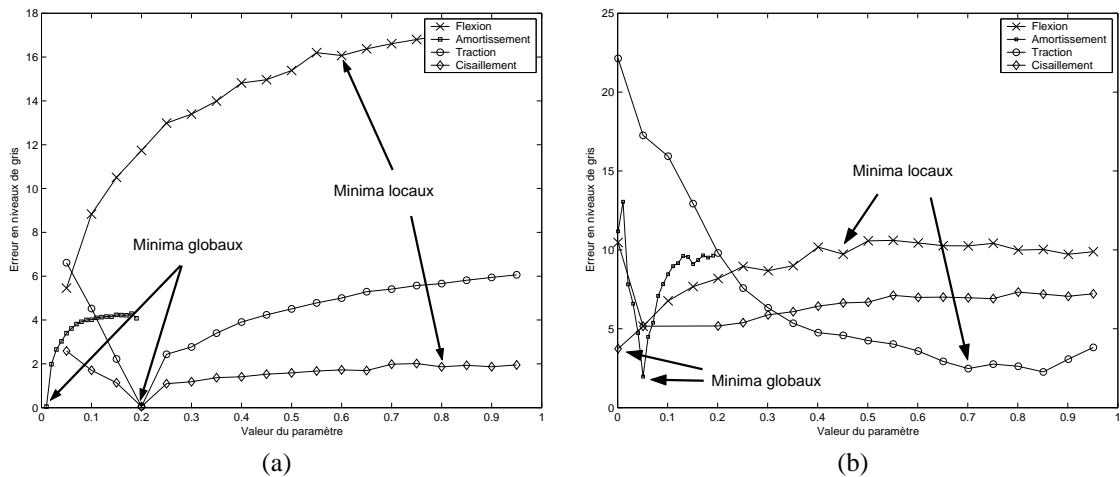


FIG. 5 – Fonctions d'erreur des paramètres dans le cas du drapé (a) et de la sphère traversant la nappe (b). On remarque qu'il y a bien un minimum global pour chaque fonction, ce qui prouve que ces fonctions sont "minimisables". On note également des minima locaux ce qui nécessite l'utilisation d'algorithmes de minimisation évolués comme le recuit simulé[16].

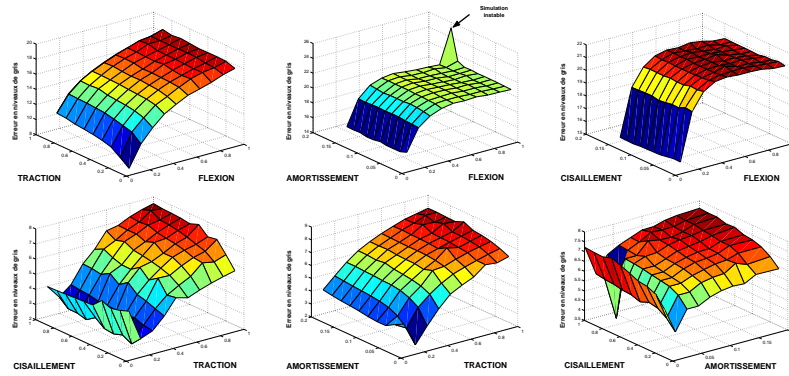


FIG. 6 – Décorrélation des paramètres dans le cas du drapé.

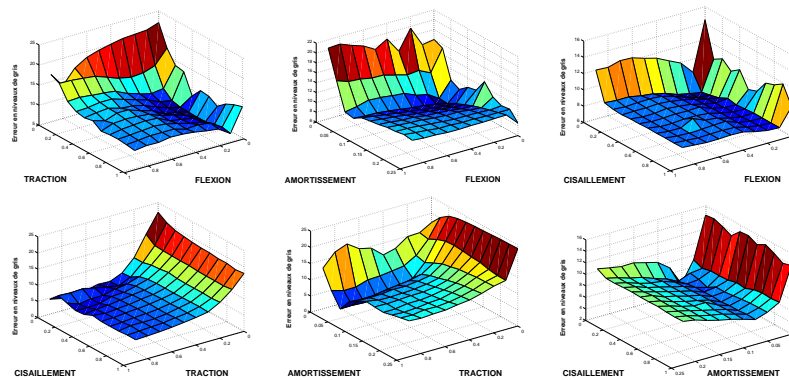


FIG. 7 – Décorrélation des paramètres dans le cas de la sphère traversant une nappe suspendue.

scènes testées<sup>6</sup>, ce qui nous permet de réaliser deux expériences d'estimation successive des propriétés physiques d'un modèle de tissu à partir de séquences d'images synthétiques.

On peut constater sur la figure 5 que l'erreur générée par certains paramètres, comme la flexion, est beaucoup plus importante que les autres. Il semble donc raisonnable de penser que minimiser l'erreur générée par la flexion permet de réduire rapidement la différence entre les deux animations. Suivant cette idée et l'observation de ces courbes, nous avons défini la hiérarchie suivante pour les deux expériences que nous avons réalisées:

1. Résistance à la flexion
2. Amortissement visqueux global
3. Résistance à la traction/compression
4. Résistance au cisaillement

Un autre avantage de l'estimation des paramètres un à un est la réduction considérable de l'espace de recherche, et donc l'accélération de la minimisation à l'aide de méthodes mono-objectifs.<sup>7</sup>

## 4 Résultats

Les tableaux TAB.1 et TAB.2 illustrent le résumé des estimations réalisées à partir des vidéos synthétiques des expériences décrites en figure 4. Nous utilisons ici la métrique définie par Bhat et al. et le recuit simulé appliqué aux paramètres un à un en utilisant la hiérarchie définie au paragraphe précédent. De plus, il est nécessaire de connaître une bonne valeur initiale du paramètre estimé pour le bon fonctionnement du recuit simulé. Nous réalisons donc, lors de l'estimation de chaque paramètre, une étude grossière de l'espace de validité (dans notre cas 5 tests) et nous choisissons comme point de départ pour la minimisation la valeur générant l'erreur la plus faible.

6. Nous n'excluons pas ici la possibilité d'une corrélation mettant en jeu plus de deux paramètres mais cela reste très peu probable.

7. Les résultats des tableaux TAB.1 et TAB.2 sont cependant obtenus avec le recuit simulé appliqué à un seul objectif. Nous envisageons



| Paramètre     | Valeur réelle | Valeur trouvée | Intervalle | Erreur | Nb Itérations | Dernière erreur |
|---------------|---------------|----------------|------------|--------|---------------|-----------------|
| Flexion       | 0.02          | 0.02834        | [0.0; 1.0] | 0.83 % | 46            | 6.31539         |
| Amortissement | 0.02          | 0.01629        | [0.0; 0.2] | 1.85 % | 96            | 5.70359         |
| Traction      | 0.20          | 0.18344        | [0.0; 1.0] | 1.66 % | 86            | 2.63123         |
| Cisaillement  | 0.20          | 0.27352        | [0.0; 1.0] | 7.35 % | 86            | 2.58639         |

314 itérations (environ 7h45 de calculs sur un bi-processeur 2.4 Go et 512 Mo de RAM)

TAB. 1 – Résultats des estimations de paramètres mécaniques depuis une séquence d’images dans le cas de la sphère traversant la nappe suspendue. L’identification des paramètres est ici de très bonne qualité. En effet, l’erreur d’estimation est minime pour tous les paramètres. L’erreur est plus importante pour le cisaillement. Cependant, cette scène n’étant pas discriminante pour ce paramètre, cette erreur n’est pas visible à l’oeil nu (inférieure à 5 niveaux de gris) à la comparaison des deux vidéos (cf fig 8). Le nombre d’itérations pourrait être réduit à l’aide d’un outil de minimisation plus spécifique (mono-objectif).

| Paramètre     | Valeur réelle | Valeur trouvée | Intervalle | Erreur  | Nb Itérations | Dernière erreur |
|---------------|---------------|----------------|------------|---------|---------------|-----------------|
| Flexion       | 0.02          | 0.20491        | [0.0; 1.0] | 18.49 % | 66            | 8.91762         |
| Amortissement | 0.05          | 0.05794        | [0.0; 0.2] | 3.97 %  | 76            | 7.90681         |
| Traction      | 0.8           | 0.99418        | [0.0; 1.0] | 19.42 % | 76            | 7.39990         |
| Cisaillement  | 0.01          | 0.11398        | [0.0; 1.0] | 10.40 % | 96            | 6.90026         |

314 itérations (environ 7h45 de calculs sur un bi-processeur 2.4 Go et 512 Mo de RAM)

TAB. 2 – Résultats des estimations de paramètres mécaniques depuis une séquence d’images dans le cas du drapé. Les erreurs d’estimation sont ici plus importantes. Le problème est que cette scène n’est pas discriminante pour la flexion qui est estimée en premier lieu. Cependant, la comparaison à l’oeil nu est acceptable (6.9 niveaux de gris de moyenne), surtout grâce à une estimation de l’amortissement correcte, d’où résulte un bon “calage” des deux vidéos (cf fig 8).

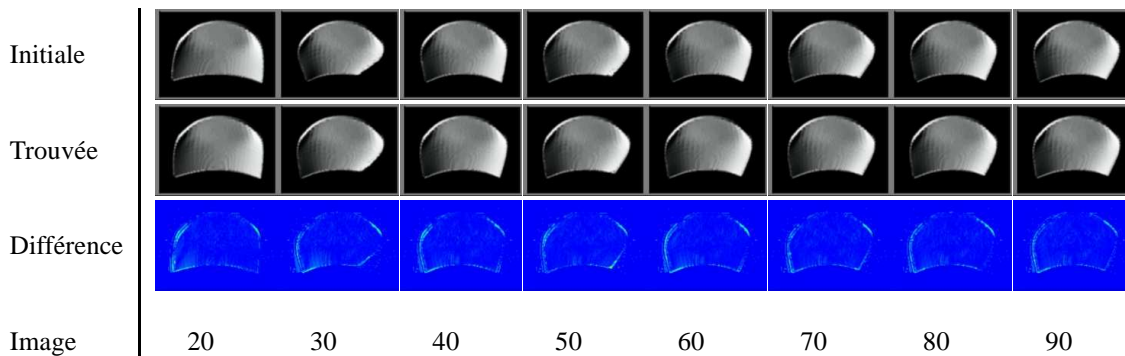


FIG. 8 – Illustration de la comparaison des cartes d’orientation locale de la vidéo initiale et de la vidéo générée à l’aide des paramètres trouvés pour le drapé. On peut voir que la différence entre les deux animations est très faible. En effet, comme indiqué sur le tableau TAB.1, l’estimation des paramètres est réalisée avec une erreur minime.

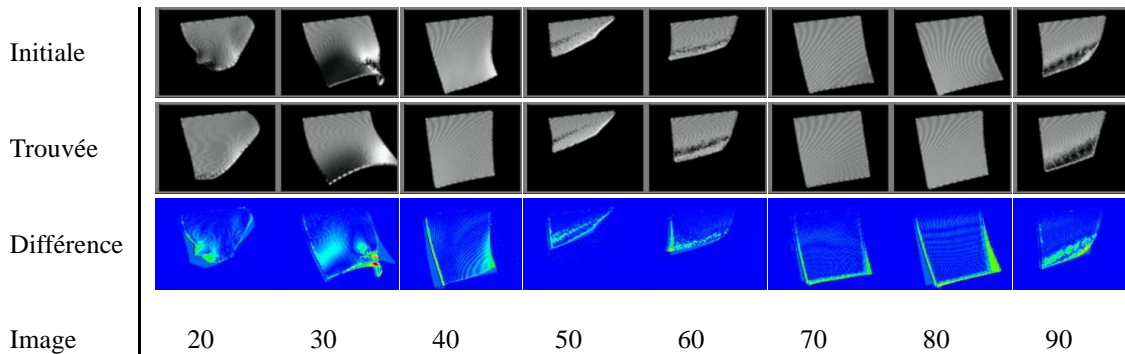


FIG. 9 – Illustration de la comparaison des cartes d’orientation locale de la vidéo initiale et de la vidéo générée à l’aide des paramètres trouvés pour la sphère traversant la nappe suspendue. Pour cette scène, la différence est un peu plus visible car l’erreur d’estimation de certains paramètres est importante (cf TAB.2). On note cependant un bon calage des deux vidéos qui est dû à une estimation relativement correcte du paramètre d’amortissement.



## 5 Conclusion et travaux futurs

Nous avons réalisé une étude de sensibilité des paramètres d'un simulateur de tissus basé sur le modèle de Provot. Nous en avons déduit une hiérarchie des paramètres que nous avons appliquée à la mécanique inverse. Nous avons ainsi obtenu des résultats prometteurs pour une estimation des paramètres de ce modèle de tissus depuis des séquences d'images synthétiques. Ceci nous permet également de valider notre approche de mécanique inverse. Cependant, il reste encore de nombreux travaux à effectuer afin de faire évoluer cette méthode vers des applications réelles. Nous citons ici quelques-unes des extensions possibles sur lesquelles nous travaillons déjà:

**Estimations à partir de vidéos réelles** Nous réaliserons l'acquisition de vidéos réelles et appliquerons la méthode sur un modèle de tissu beaucoup plus rigoureux physiquement [22]. Il sera alors sans doute nécessaire d'effectuer une estimation des paramètres photométriques en utilisant le *rendu inverse*[3] avant d'appliquer la méthode d'estimation des paramètres mécaniques.

**Accélération GPU** L'estimation reste actuellement assez lente lors de l'exécution. Bien que ce ne soit pas aujourd'hui le souci majeur (l'exactitude de l'évaluation est pour nous primordiale), cela pourrait devenir un objectif à long terme. Un facteur est la résolution du modèle géométrique. En effet, plus celle-ci est importante, plus la simulation mécanique est lente. Il pourrait alors être intéressant d'effectuer certaines de ces opérations sur le processeur de la carte graphique, pratique de plus en plus courante dans des domaines très divers [23].

**Evaluation de modèles mécaniques concurrents** Pour un type de modèle mécanique, il existe de nombreuses méthodes de simulation. L'une des conséquences directes de l'estimation est d'obtenir une valeur quantifiant l'erreur minimale obtenue avec un simulateur. Nous pensons ainsi qu'il serait possible d'utiliser l'identification à un niveau supérieur, et donc de déterminer quel simulateur est le plus adapté à une expérience donnée.

**Estimation par comparaison 3D** (cf section 2.2) Nous pensons également que la reconstruction du modèle 3D depuis la vidéo initiale pour appliquer ensuite les méthodes de comparaison d'objets géométriques [19] permettrait un gain de performances considérable. Bien que les méthodes de reconstruction ne soient pas encore adaptées aux besoins de la mécanique inverse, cette piste reste néanmoins à suivre, car la reconstruction géométrique à partir de vidéos fait l'objet de nombreuses tentatives prometteuses [26].

**Extension à d'autres simulateurs mécaniques** Nous définirons également de nouvelles métriques d'erreur qui nous permettront d'étendre le champ d'application de l'estimation de paramètres à des modèles mécaniques de natures plus variées. Nous proposons notamment pour cela d'utiliser les informations de *flux optique* [13][9]. L'idée est ici de retrouver les comportements dynamiques plutôt qu'une série de comportements statiques recherchés ici avec les cartes d'orientations locales. Par exemple, dans le cas du tissu, nous pensons qu'il serait plus judicieux de baser la ressemblance des vidéos sur le nombre de plis total formés au cours de l'animation plutôt que la recherche sur quelques plis à des moments bien précis de celle-ci.

Nous désirons ainsi étendre la méthode de la *mécanique inverse* (adjointe au rendu inverse) à un ensemble de modèles mécaniques beaucoup plus vaste, notamment dans le domaine de la simulation chirurgicale, pour retrouver par exemple les propriétés physiques d'un oeil, d'un foie [24], ou encore de fils chirurgicaux [18].

## Références

- [1] D. Baraff and A. Witkin. Large steps in cloth simulation. In *Proceedings of ACM SIGGRAPH 98*, pages 43–54. ACM Press, 1998.
- [2] K. Bhat, C. Twigg, J. Hodgins, P. Khosla, Z. Popović, and S. Seitz. Estimating cloth simulation parameters from video. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2003)*, pages 37–51. ACM Press, 2003.
- [3] Samuel Boivin and Andre Gagalowicz. Image-based rendering of diffuse, specular and glossy surfaces from a single image. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 107–116. ACM Press, 2001.
- [4] Richard P. Brent. *Algorithms for Minimization without Derivatives*. Prentice-Hall, Englewood Cliffs, NJ, 1973.
- [5] R. Bridson, R. Fedkiw, and J. Anderson. Robust treatment of collisions, contact and friction for cloth animation. *ACM Transactions on Graphics (ACM SIGGRAPH 2002)*, 21(3):594–603, July 2002.

- [6] R. L. Carceroni and K. N. Kutulakos. Multi-view scene capture by surfel sampling: From video streams to nonrigid 3d motion, shape & reflectance. In *International conference on Computer Vision*, pages 60–67, 2001.
- [7] Robert L. Cook and Kenneth E. Torrance. A reflectance model for computer graphics. *SIGGRAPH, Computer Graphics Proceedings*, 15:301–316, July 1981.
- [8] William T. Freeman and Edward H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(0):891–906, 1991.
- [9] D. Gibson and M. Spann. Robust optical flow estimation based on a sparse motion trajectory set. *IP*, 12(4):431–445, April 2003.
- [10] J. Haddon, D. Parks, and D. Forsyth. The appearance of clothing, 1998.
- [11] John A. Haddon and David A. Forsyth. Shading primitives: Finding folds and shallow grooves. In *ICCV*, pages 236–241, 1998.
- [12] Berthold K. P. Horn and Michael J. Brooks. *Shape from shading*. MIT Press, 1989.
- [13] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.
- [14] Henrik Wann Jensen. Global illumination using photon maps. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, pages 21–30. Springer Verlag, 1996.
- [15] A. Joukhadar, F. Garat, and Ch. Laugier. Parameter identification for dynamic simulation.
- [16] Kirkpatrick, Gelatt, and Vecchi. Optimization by simulated annealing. *Science*, pages 220:671–680, 1983.
- [17] Jan J. Koenderink, Andrea J. van Doorn, Kristin J. Dana, and Shree K. Nayar. Bidirectional reflection distribution function of thoroughly pitted surfaces. *International Journal of Computer Vision*, 31(2/3):129–144, July 1999.
- [18] Julien Lenoir, Philippe Meseure, Laurent Grisoni, and Christophe Chaillou. Surgical thread simulation. In Marc Thiriet, editor, *Modelling and Simulation for Computer-aided Medecine and Surgery (MS4CMS)*, volume 12, pages 102–107, Rocquencourt (France), 12-15 nov 2002. INRIA, EDP Sciences.
- [19] Jean Louchet, Xavier Provot, and David Crochemore. Evolutionary identification of cloth animation models. In Dimitri Terzopoulos and Daniel Thalmann, editors, *Computer Animation and Simulation '95*, pages 44–54. Springer-Verlag, 1995.
- [20] Philippe Meseure. Animation basée sur la physique pour les environnements interactifs temps-réel. Habilitation à Diriger des Recherches, 17 décembre 2002.
- [21] Philippe Meseure, Jérôme Davanne, Laurent Hilde, Julien Lenoir, Laure France, Frédéric Triquet, and Christophe Chaillou. A physically-based virtual environment dedicated to surgical simulation. In *Surgery Simulation and Soft Tissue Modeling (IS4TM'03)*, pages 38–47, Juan-les-pins, jun 2003.
- [22] C. Ngo Ngoc and S. Boivin. Nonlinear cloth simulation. Research report, INRIA Futurs, 2004.
- [23] Timothy J. Purcell, Craig Donner, Mike Cammarano, Henrik Wann Jensen, and Pat Hanrahan. Photon mapping on programmable graphics hardware. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, pages 41–50. Eurographics Association, 2003.
- [24] Kenneth Sundaraj. *Real-Time Dynamic Simulation and 3D Interaction of Biological Tissue: Application to Medical Simulators*. Thesis / phd, GRAVIR / IMAG, INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE - INPG, 2004.
- [25] C. Syllebranque. Estimation de propriétés mécaniques d'un modèle de tissus à partir de séquences d'images. Master's thesis, Université des Sciences et Technologies de Lille, 2004.
- [26] Lorenzo Torresani, Aaron Hertzmann, and Christoph Bregler. Learning non-rigid 3d shape from 2d motion. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.
- [27] Yizhou Yu, Paul Debevec, Jitendra Malik, and Tim Hawkins. Inverse global illumination: recovering reflectance models of real scenes from photographs. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 215–224. ACM Press/Addison-Wesley Publishing Co., 1999.