

Éléments finis robustes pour l'animation interactive de solides déformables

Matthieu Nesme^{1,2}, François Faure¹, Matthieu Chabanas²

¹EVASION-GRAVIR INRIA ²GMCAO-TIMC IMAG
prenom.nom@imag.fr

Résumé : *Cet article présente un modèle physique, simple et robuste, d'objets viscoélastiques basé sur la méthode des éléments finis. Récemment, plusieurs méthodes permettant d'éliminer les artefacts inhérents au calcul linéaire des déformations en grands déplacements ont été proposées. Leur idée principale est de décomposer les déplacements en un mouvement rigide et des déformations pures. Dans cet article nous présentons une nouvelle décomposition rapide et robuste, qui ne compromet pas le réalisme physique. De plus nous montrons comment nous mettons à jour, à faible coût, la matrice de rigidité permettant un meilleur réalisme physique en grandes déformations. Notre approche permet d'animer environ 2000 tétraèdres en temps-réel.*

Mots-clés : animation, modèle physique, élasticité, éléments finis

1 Introduction

L'animation par modèles physiques s'est révélée indispensable dans le domaine de l'informatique graphique. L'élaboration de simulateurs sous contraintes de calculs temps réel et avec des objectifs de « réalisme » comportemental est devenue une problématique centrale. En effet, cette communauté cherche maintenant à étendre ses outils de modélisation vers des méthodes mathématiques plus proches de la théorie qui décrit la mécanique des milieux continus (du « réaliste » vers le « précis »), et ceci avec un souci nouveau de confrontation des modèles avec des données issues du monde réel. Pour cela, il faut être capable de simuler avec réalisme les déformations de structures complexes, tout en conservant une rapidité compatible avec une utilisation interactive.

Depuis que Terzopoulos a simulé des corps déformables viscoélastiques et plastiques [TPBF87, TF88], il a été développé un grand nombre de méthodes basées sur la méthode des éléments finis ou des différences finies.

Deux grandes écoles travaillent parallèlement, ceux qui essaient de réaliser les modèles les plus réalistes, avec le maximum d'effets afin de refléter au mieux la réalité et ceux qui essaient d'avoir le meilleur compromis réalisme physique / temps de calcul, avec le souci d'interactivité. Nous nous intéressons à ces dernières méthodes.

Dans un souci de rapidité des calculs, la plupart des méthodes interactives utilisent un calcul linéaire des déformations, comme [CDC⁺96] et les systèmes masses-tenseurs [CDA00, PDA00] de Cotin et de Delingette. Malheureusement ces modèles ne fonctionnaient qu'en petits déplacements, c'est pourquoi Pincinbonno et Debunne ont utilisé un calcul non-linéaire des déformations [PDA03, DCB01, DDCB00, DDBC99]. Récemment, Müller [MDM⁺02, MTG04, MG04], Eitzmuß [EK03] et Irving [ITF04] ont préféré conserver un tenseur linéaire en offrant la possibilité de simuler des matériaux purement linéaires, tout en gérant les grands déplacements. Malheureusement, ces dernières combinent difficilement réalisme physique et vitesse. C'est pour cette raison, que nous étendons ces méthodes avec un calcul plus rapide et robuste des déformations.

2 Modèle déformable

On s'intéresse à des corps viscoélastiques qui tendent à reprendre une forme de référence dite « forme au repos ». Pour simuler de tels corps, nous devons : modéliser les déformations, en déduire des actions mécaniques et traiter les aspects temporels pour des applications dynamiques.

2.1 Cinématique des milieux continus

Nous savons que la matière est discontinue à l'échelle moléculaire, mais à notre échelle macroscopique, elle se présente comme un milieu continu, c'est à dire dont les propriétés physiques varient continuellement d'un point à

un autre. On s'intéresse à trouver les forces qui s'exercent sur un objet qui a été déformé, et réciproquement les déformations résultantes aux forces exercées sur l'objet.

2.1.1 Déplacements

Sous l'action de forces, un corps non rigide se déforme et se déplace. On peut définir la position initiale d'un point M de ce corps par son vecteur position x_{repos} . Lorsque le corps est déformé, chacun de ses points se retrouve à une position différente $x_{deforme}$. Le déplacement u correspond au vecteur entre l'état déformé et l'état au repos d'une particule : $u(M) = x_{deforme}(M) - x_{repos}(M) = (u_x(M) \ u_y(M) \ u_z(M))^T$.

2.1.2 Déformations

A partir des déplacements, on est en mesure de connaître l'état de déformation de l'objet en une particule, représenté par le tenseur de déformations. Pour cela, le calcul de ce tenseur fait intervenir la géométrie du milieu. Les déformations représentent les dilatations (élongation, compression) et les cisaillements que subit un objet. Il existe plusieurs calculs des déformations.

Pour être insensible aux translations, les déformations se calculent en considérant le gradient du déplacement $[grad\ u]$ car $u(M + dM) = u(M) + [grad\ u] dM$ avec

$$grad\ u = \begin{bmatrix} \frac{\partial u_x}{\partial x} & \frac{\partial u_x}{\partial y} & \frac{\partial u_x}{\partial z} \\ \frac{\partial u_y}{\partial x} & \frac{\partial u_y}{\partial y} & \frac{\partial u_y}{\partial z} \\ \frac{\partial u_z}{\partial x} & \frac{\partial u_z}{\partial y} & \frac{\partial u_z}{\partial z} \end{bmatrix}$$

Le tenseur de Cauchy $[\varepsilon_C] = \frac{1}{2}([grad\ u] + [grad^T\ u])$ utilise la décomposition 2.1 pour trouver les déformations pures.

$$[grad\ u] = \underbrace{\frac{1}{2}([grad\ u] + [grad^T\ u])}_{de\ formation\ pure} + \underbrace{\frac{1}{2}([grad\ u] - [grad^T\ u])}_{\approx\ petite\ rotation} \quad (2.1)$$

L'avantage de ce tenseur est son insensibilité aux translations et son calcul simple car linéaire. Son inconvénient est qu'il est invariant en rotation seulement dans le cas de petites rotations.

Lorsque les déplacements augmentent, en particulier lorsqu'une partie du système subit une rotation, on s'éloigne du vrai comportement, on devient alors de plus en plus imprécis. On est donc obligé de travailler seulement sur des objets qui ne tournent pas et se déforment peu. Cette inexactitude se caractérise par l'apparition de certains artefacts en grands déplacements, car les particules ne peuvent se déplacer que le long de trajectoires droites, visuellement, le modèle linéaire semble faire gonfler l'objet.

Pour ces raisons, le tenseur de déformations couramment utilisé pour représenter un objet élastique qui se déforme beaucoup est celui de Green-Lagrange : $[\varepsilon_G] = \frac{1}{2}([grad\ u] + [grad\ u]^T + [grad\ u]^T[grad\ u])$.

Le gros avantage de Green-Lagrange est qu'il garanti plus de liberté dans les mouvements en autorisant des déplacements en courbe qui conservent la forme de l'objet, même dans le cas de grandes rotations.

Si on veut animer des objets se déplaçant et se déformant beaucoup, l'utilisation du tenseur de déformation de Green-Lagrange semble inévitable. Or, ce tenseur non-linéaire a aussi un inconvénient qui réside justement dans son terme quadratique, car il ne permet pas de simuler des objets purement élastiques linéaires.

2.1.3 Comportement viscoélastique

Dans ce document, nous nous limitons au cas des matériaux élastiques linéaires isotropes par souci de simplicité, mais l'extension à des matériaux plus complexes est possible. Grâce à la loi de Hooke, nos matériaux se caractérisent par seulement deux paramètres : le module d'Young E correspondant à la rigidité du matériau, et le coefficient de Poisson ν correspond à son incompressibilité.

2.2 Méthodes des éléments finis

Il est souvent impossible de trouver analytiquement la solution à un problème de la mécanique des milieux continus. On doit alors utiliser des méthodes numériques pour approcher cette solution. La méthode des éléments finis (MEF) est une de ces méthodes, elle est la plus utilisée dans le cas de calcul sur des structures.

La méthode des éléments finis est un outil de discrétisation, où l'idée est de découper l'objet en un nombre fini d'éléments, dont les sommets sont appelés noeuds. Les propriétés physiques sont interpolables sur chaque élément en fonction de leur valeur aux noeuds. Les équations mécaniques du système doivent être vérifiées sur chacun des éléments, procurant des équations sur les noeuds. Il ne suffit pas de prendre en compte le cas d'un seul élément isolé. Le calcul des forces se fait élément par élément avec des systèmes locaux. Or dans un maillage d'éléments finis, un noeud donné appartient en général à plusieurs éléments et la déformation de chacun d'entre eux induit une force en ce noeud.

Dans le cas d'une résolution statique du problème qui a pour vocation de calculer directement une configuration d'équilibre, on regroupe généralement tous les systèmes locaux en un système global assemblé contenant les équations de tous les noeuds. Cette technique nécessitant l'inversion d'un système, est lourde en calcul, elle est donc peu adaptée aux problèmes de dynamique. De plus, la présence d'un système global, rend les changements de topologie très difficiles. C'est un point important, que l'on n'a pas traité mais qu'il faut prendre en compte car il peut être important de simuler des déchirures et des découpes, surtout dans le cadre chirurgical.

Puisque dans le cas de problèmes dynamiques, on ne cherche pas à trouver la position d'équilibre du système global, mais à calculer les accélérations des sommets, il est possible de calculer les forces qui s'exercent sur chaque sommet pour chaque élément indépendamment. Ensuite, on somme toutes les forces qui s'appliquent à un sommet auquel est associée une masse. Ce type de résolution dite "explicite" fait gagner énormément en vitesse de calcul, mais plusieurs pas de calcul sont nécessaires pour propager les déformations, chaque sommet ne réagissant qu'à la position de ses voisins à l'instant précédent. Par contre l'état final d'équilibre du système amorti est le même que calculé par la statique (à un déplacement rigide près).

Un autre avantage de ce type de résolution est de faciliter la combinaison de plusieurs types d'éléments dans le maillage, puisque chaque élément est traité indépendamment.

2.2.1 Calcul linéaire

Dans cette partie, nous allons voir la formulation linéaire des tenseurs qui rend l'équation plus facile à implémenter.

Pour chaque élément, on a :

$$f_{deforme \rightarrow repos} = Ku = B^T DB(x_{repos} - x_{deforme}) \quad (2.2)$$

– B est la matrice de rigidité de l'élément :

$$B = \begin{bmatrix} \frac{\partial N_1}{\partial x} & 0 & 0 & \dots & \frac{\partial N_n}{\partial x} & 0 & 0 \\ 0 & \frac{\partial N_1}{\partial y} & 0 & \dots & 0 & \frac{\partial N_n}{\partial y} & 0 \\ 0 & 0 & \frac{\partial N_1}{\partial z} & \dots & 0 & 0 & \frac{\partial N_n}{\partial z} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_1}{\partial x} & 0 & \dots & \frac{\partial N_n}{\partial y} & \frac{\partial N_n}{\partial x} & 0 \\ 0 & \frac{\partial N_1}{\partial z} & \frac{\partial N_1}{\partial y} & \dots & 0 & \frac{\partial N_n}{\partial z} & \frac{\partial N_n}{\partial y} \\ \frac{\partial N_1}{\partial z} & 0 & \frac{\partial N_1}{\partial x} & \dots & \frac{\partial N_n}{\partial z} & 0 & \frac{\partial N_n}{\partial x} \end{bmatrix}$$

où N_i est la fonction d'interpolation du i^{ime} noeud et n le nombre de noeuds de l'élément ; les fonctions d'interpolation dépendent du type de l'élément.

– D est la matrice de rigidité du matériau :

$$D = \varphi(V) \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1 & \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\ \frac{\nu}{1-\nu} & 1 & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\ \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} \end{bmatrix}$$

où $\varphi(V)$ est une fonction décroissante du volume et dépendant du type d'élément.

– $x = (x_{1_x} \ x_{1_y} \ x_{1_z} \ \dots \ x_{n_x} \ x_{n_y} \ x_{n_z})^T$, $u = (u_{1_x} \ u_{1_y} \ u_{1_z} \ \dots \ u_{n_x} \ u_{n_y} \ u_{n_z})^T$ et $f_{deforme \rightarrow repos} = (f_{1_x} \ f_{1_y} \ f_{1_z} \ \dots \ f_{n_x} \ f_{n_y} \ f_{n_z})^T$ représentent respectivement les positions, les déplacements et les forces appliquées aux n noeuds d'un élément.

Bu correspond aux déformations : $\{\varepsilon\} = Bu$

DBu correspond aux contraintes : $\{\sigma\} = DBu = D\{\varepsilon\}$

$K = B^TDB$ est appelée matrice de rigidité.

2.3 Dynamique

Dans le cas de simulations interactives, on ne cherche pas explicitement un état d'équilibre, mais plutôt des déformations au cours du temps.

Si on veut visualiser et interagir à tout moment sur le système, il faut faire intervenir le temps et gérer le problème en dynamique. Pour cela, on doit résoudre une équation différentielle du second ordre de la forme :

$$M\ddot{u} + C\dot{u} + Ku = f$$

où M est lié à la masse, C à l'amortissement (on peut utiliser $C = \alpha K$) et K à la rigidité, u correspond aux déplacements et f aux forces ; le temps est traité comme un paramètre. M , C et K peuvent dépendre de la géométrie et donc devoir être recalculées à chaque déformation.

On résout donc un problème à chaque pas de temps, en partant d'un instant initial où les positions et vitesses sont supposées connues : ce sont les conditions initiales.

Le schéma d'intégration est l'algorithme qui permet de calculer l'état à l'instant suivant à partir de l'instant courant en effectuant une intégration du temps, comme exprimé par la formule :

$$x(t+h) = x(t) + \int_t^{t+h} \dot{x}(x,t) dt$$

où $x(t)$ est la position d'une particule au temps t , $\dot{x}(x,t)$ est sa vitesse et h est le pas de temps.

Il existe quantité de schémas d'intégration, qui se distinguent par leur ordre de précision, leur stabilité et leur coût en terme de temps de calcul. Tous sont approximatifs car on ne sait pas calculer l'intégrale de la formule dans le cas général.

La méthode qui a été retenue comme référence pour notre travail est celle d'Euler implicite. Il semble que les intégrations basées sur cette méthode offrent un bon compromis stabilité / vitesse / précision.

Notre intégration dynamique se formule de la manière suivante :

$$\begin{cases} x(t+h) = x(t) + h\dot{x}(t+h) = x(t) + \Delta x \\ \dot{x}(t+h) = \dot{x}(t) + h\ddot{x}(t+h) = \dot{x}(t) + \Delta \dot{x} \end{cases}$$

Grâce au développement donné par Baraff dans [BW98], ce système est résoluble par la méthode des gradients conjugués, et pour cela, il faut pouvoir calculer la variation de la force à partir de la variation de la position. La suite de cet article donne une nouvelle façon de calculer $[\frac{\partial f}{\partial x}] \Delta \dot{x}$. Vu que l'on développe une MEF explicite, cette matrice $[\frac{\partial f}{\partial x}]$ n'est pas globalement construite, mais pour chaque noeud, on somme les contributions de chaque élément.

2.4 Déformations linéaires et grands déplacements

Lorsque l'on utilise un tenseur de déformations linéaire, il faut pouvoir distinguer les déplacements rigides (translation et rotation) des déformations pures (dilatation, cisaillement) afin de ne pas avoir les artefacts (gonflements)

liés aux grands déplacements. Pour cela, il faut éliminer les déplacements rigides et ne prendre en compte que les déformations pures pour calculer les forces exercées.

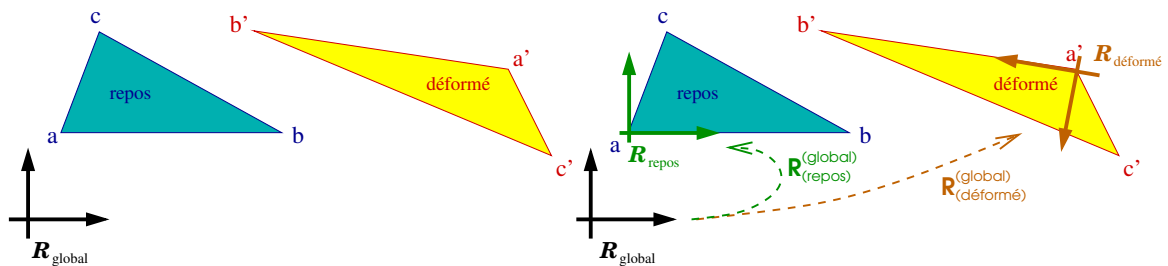
Nous partons du principe que puisque le tenseur de Cauchy est déjà invariant en translation, il suffit d'être invariant en rotation. La méthode [MDM⁺02] qui travaille sur les noeuds ne donne pas des résultats satisfaisants, car elle crée des forces fantômes, c'est pourquoi on se tourne vers les méthodes travaillant sur les éléments. L'idée globale de ces méthodes considérées [EK03, MG04, ITF04] est de décomposer la transformation de l'élément déformé vers l'élément au repos afin d'en extraire la rotation, ainsi que les déformations subies par l'élément.

Il n'existe pas un seul "état de rotation" commun entre un élément au repos et déformé, ce qui est important, c'est d'en trouver un qui permette de trouver des déformations résultant de petits déplacements. Les décompositions, polaire dans [EK03, MG04] et SVD dans [ITF04], permettent de trouver les plus petites déformations, celles qui approcheront au mieux la vraie loi de comportement. Mais ces méthodes reposent sur des algorithmes relativement coûteux.

Nous présentons une autre décomposition aux calculs plus rapides, se prêtant mieux aux simulations interactives. Celle-ci ne donne pas les déformations basées sur les plus petits déplacements, mais en donne quand même à partir de petits déplacements. La précision est alors du même ordre qu'une méthode classique utilisant un tenseur de déformations linéaire en petits déplacements.

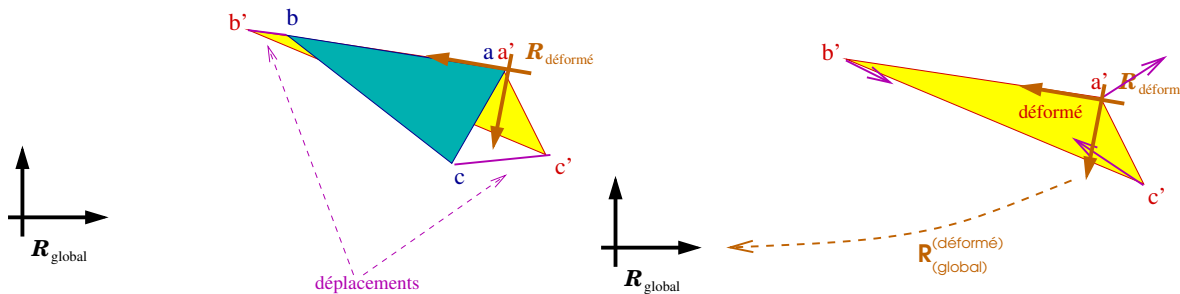
3 Notre méthode

3.1 Invariance aux rotations



(a) étape 1 : l'élément initialement au repos (vert) est maintenant déplacé et déformé (jaune).

(b) étape 2 : on calcule les rotations du repère global vers les repères locaux, et on ramène les coordonnées de chaque sommet dans le repère local de leur élément.



(c) étape 3 : avec ces coordonnées locales, on peut mesurer des petits déplacements.

(d) étape 4 : les forces calculées dans le repère déformé sont ramenées dans le repère global.

FIG. 1 – Calcul des forces avec des petits déplacements après élimination des rotations

Les méthodes présentées précédemment trouvent directement les déformations pures en décomposant les déplacements contenant aussi une rotation. Nous adaptons le problème, en ne cherchant pas directement les déformations, mais en cherchant seulement la rotation. Celle-ci nous permettant de ramener l'élément au repos dans un même "état de rotation" que l'élément déformé. Dans ce repère il sera alors possible de mesurer de petits déplacements, avec lesquels un calcul linéaire des déformations est suffisant.

Nous ne cherchons pas directement la rotation $R_{(repos)}^{(deforme)}$ entre l'élément déformé et l'élément au repos, mais nous décomposons le problème en passant par un repère intermédiaire : le repère (*global*). L'utilisation de ce repère, en plus de simplifier les calculs, semble inévitable puisqu'au départ nous ne connaissons que les coordonnées globales. Notre problème se réduit donc à trouver les rotations $R_{(repos)}^{(global)}$ et $R_{(global)}^{(deforme)}$ (figure 1(b)) : $R_{(repos)}^{(deforme)} = R_{(repos)}^{(global)} R_{(global)}^{(deforme)}$.

En opposition au repère (*global*), les repères (*repos*) et (*deforme*) sont appelés repères "locaux".

L'idée est de mesurer des petits déplacements entre les coordonnées exprimées dans les repères locaux, pour lesquels les éléments sont dans un même "état de déformation" (figure 1(c)). Pour cela, il faut que ces repères ne dépendent que de l'élément lui-même. Pour simplifier au maximum les calculs, les repères choisis sont orthonormés directs. Leur centre correspond à un sommet de l'élément, et une arête adjacente normalisée fait office de premier axe. Les autres axes sont déduits par construction, le second axe étant dans le plan formé par le premier axe et une autre arête adjacente au centre du repère. Bien sûr, pour un élément, ce sont toujours les mêmes sommets qui doivent être utilisés pour construire des repères comparables.

Grâce à la simplicité des repères locaux, il est aisé de calculer la rotation qui amène les coordonnées connues dans le repère (*global*) vers les repères locaux.

Par exemple, en considérant un tétraèdre formé des points de positions a, b, c et d définies dans le repère (*global*), la rotation $R_{(local)}^{(global)}$ autour du point de position a amenant un point défini dans le repère (*global*) de l'élément vers le repère (*local*) est :

$$R_{(local)}^{(global)} = (R_{(global)}^{(local)})^T = [(1) \quad (2) \quad (3)]^T$$

$$(1) = \frac{ab}{\|ab\|} \quad (3) = \frac{ab \wedge ac}{\|ab \wedge ac\|} \quad (2) = (1) \wedge (3)$$

On peut voir que le calcul de la rotation dépend uniquement de 3 points, et devrait donc pouvoir s'appliquer à tous types d'éléments. Il n'a été utilisé que sur des triangles et des tétraèdres, mais on peut imaginer le tester sur des hexaèdres, ce qui serait très intéressant quand on sait que ceux-ci offrent de meilleurs résultats dans la simulation de solides élastiques [Gar02].

Grâce à ces changements de repères, nous sommes maintenant en possession de petits déplacements mesurés dans le repère (*local*). A partir de ces déplacements, nous sommes en mesure de calculer les forces en chacun des sommets. Les forces sont donc calculées dans le repère (*deforme*), soit à la rotation inverse $R_{(global)}^{(deforme)}$ près, il faut donc les ramener dans le repère (*global*) (figure 1(d)).

Notre formule 2.2 de calcul des forces de réponse f en chaque sommet d'un élément va donc devenir de la forme :

$$\begin{aligned} f_{deforme \rightarrow repos}^{(global)} &= RR_{(global)}^{(deforme)} K (x_{repos}^{(repos)} - x_{deforme}^{(deforme)}) \\ &= RR_{(global)}^{(deforme)} K (RR_{(repos)}^{(global)} x_{repos}^{(global)} - RR_{(deforme)}^{(global)} x_{deforme}^{(global)}) \end{aligned} \quad (3.1)$$

où $RR_{(i)}^{(j)}$ est une matrice (carrée de largeur le nombre de sommets de l'élément) contenant en diagonale la rotation $R_{(i)}^{(j)}$ du repère j vers le repère i . Par exemple, pour un tétraèdre, soit 4 sommets :

$$RR_{(i)}^{(j)} = \begin{bmatrix} R_{(i)}^{(j)} & 0 & 0 & 0 \\ 0 & R_{(i)}^{(j)} & 0 & 0 \\ 0 & 0 & R_{(i)}^{(j)} & 0 \\ 0 & 0 & 0 & R_{(i)}^{(j)} \end{bmatrix}_{12 \times 12}$$

En plus de faciliter son calcul, un avantage de décomposer la rotation $R_{(deforme)}^{(repos)}$, est que la rotation $R_{(repos)}^{(global)}$ peut

être pré-calculée, et on peut, avant de commencer l'animation, déjà avoir les coordonnées locales des sommets de l'élément au repos $x_{repos}^{(repos)}$.

3.2 Mise à jour de la matrice de rigidité

La rigidité K dépend à la fois du matériau D et de l'élément B ($K = B^T D B$). La matrice de rigidité du matériau D , ne dépendant que du matériau et de la géométrie au repos, peut être précalculée. Il faut noter que dans notre modèle, nous faisons intervenir uniquement le volume initial de l'élément pour le calcul de la rigidité. En effet, il ne semble pas logique qu'un élément devienne plus mou parce qu'il est dilaté. La rigidité de l'élément B dépend de sa forme, son calcul revient à calculer les fonctions d'interpolations. Lors de déformations, la géométrie change et cette interpolation varie, il est donc important de la recalculer à chaque pas de temps pour chaque élément.

Classiquement, le calcul de cette matrice prend un peu de temps. C'est à cause de cette lourdeur des calculs qu'à notre connaissance, actuellement aucune méthode temps-réel ne peut se permettre de mettre à jour ces matrices de rigidité, et qu'elles sont précalculées sur l'élément au repos.

Or cette simplification, si elle n'entraîne pas de forces fantômes, peut entraîner des moments fantômes. En effet, même si la somme des forces est nulle sur un élément, il se peut que celui-ci subisse un couple dû aux variations d'écartements entre les points d'application des forces. Le système n'est alors plus physiquement correct. Dans le cas où les forces sont calculées avec la matrice de rigidité de l'élément au repos, les forces fournies donnent un moment nul sur l'élément au repos, mais ne vont pas donner le même moment, et vont donc créer un couple sur l'élément déformé. Notons que nous ne cherchons pas les forces qui amènent un élément au repos vers son état déformé, mais le contraire, or $f_{deforme \rightarrow repos} \neq -f_{repos \rightarrow deforme}$ car la géométrie de l'élément déformé n'est pas la même que celle de l'élément au repos, donc l'application des mêmes forces n'aura pas le même comportement.

Grâce à des simplifications apportées par notre repère local, on est en mesure d'accélérer ce calcul. On sait que les coordonnées du premier point a sont nulles dans le repère local. De plus, par construction de nos repères, on sait que le deuxième sommet b se trouve sur l'axe \vec{Ox} , que le troisième point c est dans le plan (\vec{Ox}, \vec{Oy}) . Grâce à ces nombreux termes nuls, le calcul de la matrice de rigidité se trouve simplifié. Il est alors possible de la recalculer à chaque pas de temps pour un coût faible.

Pour un tétraèdre (a, b, c, d) , les coefficients des fonctions de forme du type $N_i = \alpha_i + \beta_i x + \gamma_i y + \delta_i z$ deviennent :

$$\begin{aligned} \beta_1 &= -c_y d_z & \beta_2 &= c_y d_z & \beta_3 &= 0 & \beta_4 &= 0 \\ \gamma_1 &= (c_x d_z) - (b_x d_z) & \gamma_2 &= c_x d_z & \gamma_3 &= d_z b_x & \gamma_4 &= 0 \\ \delta_1 &= c_y d_x - c_x d_y + b_x d_y - b_x c_y & \delta_2 &= c_y d_x - c_x d_y & \delta_3 &= -d_y b_x & \delta_4 &= -b_x c_y \end{aligned}$$

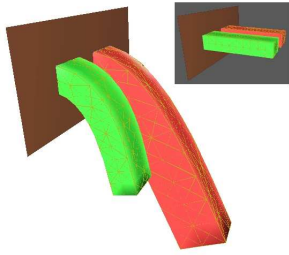
Grâce à cette prise en compte de la nouvelle forme de l'objet, notre modèle garantit une réalité physique, et une bonne stabilité.

4 Résultats

4.1 Conservation de volume

Notre méthode est conçue pour fonctionner en grands déplacements sans artefacts, il est important de vérifier que le volume ne varie pas trop quelques soient les déformations (même si dans un corps élastique, il n'y a pas une conservation exacte du volume avec la prise en compte du coefficient de Poisson). Il peut être intéressant de faire une moyenne de tous les volumes à chaque pas de temps de l'état de départ, jusqu'à l'état d'équilibre, pour vérifier que lors de l'intégration dynamique, on reste aussi correct.

Les résultats fournis figure 2 sont issus d'une simulation de poutres encastrees (mêmes maillages, mêmes paramètres) soumises à la gravité.

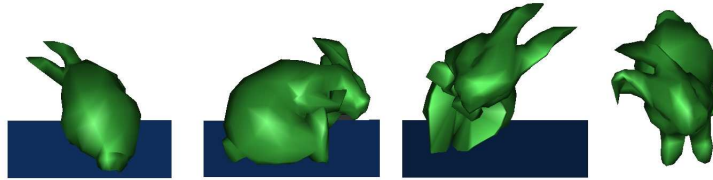
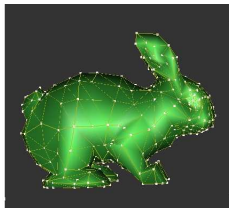


SOMME DES VOLUMES DES ÉLÉMENTS	sans gestion des grands déplacements	avec gestion des grands déplacements
en position de départ	360	360
à l'état d'équilibre	1022	362
moyenne au cours de l'animation	794	361

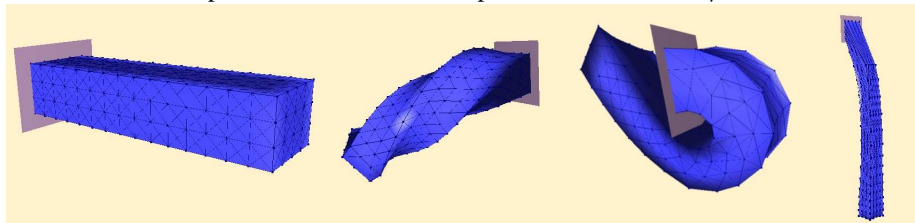
FIG. 2 – Variation du volume d'une poutre encastrée

4.2 Robustesse

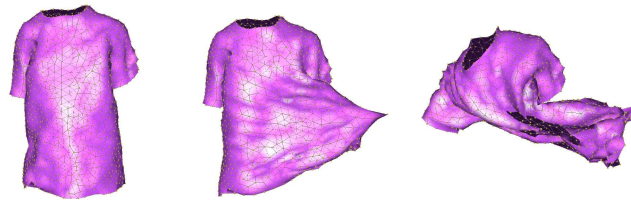
La figure 3 montre le comportement de notre méthode face à de grosses rotations de certaines parties des objets. Les images de gauche représentent les éléments au repos, et celles de droite sont des états après déformation. On voit également sur la dernière image de poutre (où l'importante déformation est due à une énorme gravité) que quelque soit l'état de déformation de l'objet, le simulateur reste stable. Notre méthode fonctionne aussi bien sur des objets volumiques (lapin, poutre) que sur des objets surfaciques (T-shirt).



Le lapin, 2791 tétraèdres, 730 particules, $E=20000$, $\mu=0.3$



La poutre, 1800 tétraèdres, 320 particules



Le T-shirt, 4266 triangles, 2181 particules, $E=5000$, $\mu=0.3$

FIG. 3 – Déformations induisant de fortes rotations des éléments

La figure 4 montre l'allure des forces en fonction des déplacements pour notre méthode. Ce qui est intéressant de noter est que dès que l'élément entre en phase d'inversion, les contraintes sont toujours négatives, cherchant toujours à ramener l'élément dans sa position initiale. Ceci garanti une grande stabilité quelque soit l'état de déformation de l'élément, même dans le cas d'une inversion. Cette dégénérescence n'est pas à proprement parlé physique, mais il est important de pouvoir y faire face pour garantir la stabilité du simulateur [ITF04].

La figure 5 donne l'exemple le pire dans le cas de dégénérescence et d'inversion, puisque tous les éléments se retrouvent dans un même plan. Notre méthode reste très stable et réagit logiquement face à ce cas.

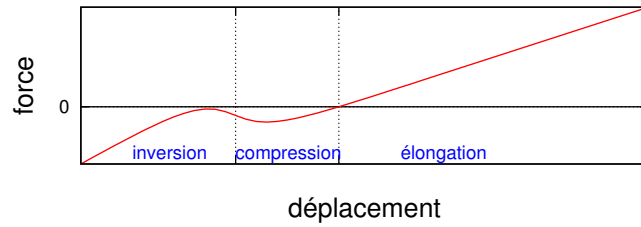


FIG. 4 – Relation entre déplacement et force pour notre méthode

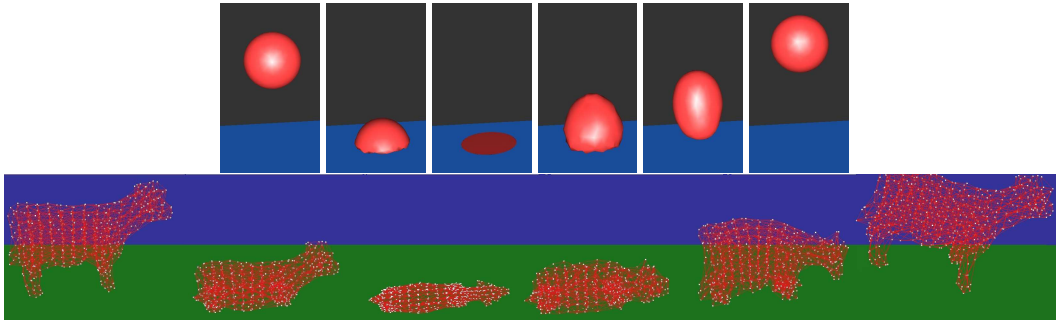


FIG. 5 – Des objets de raideur nulle (module d'Young à 0) s'aplatissent complètement sur le sol. En augmentant leur raideur, ils retrouvent leur forme initiale.

4.3 Vitesse

Les chiffres qui sont donnés ici, même s'ils sont prometteurs, résultent d'un code optimisable, on peut donc espérer les revoir à la hausse. Le plus intéressant dans ces chiffres, c'est de voir que la vitesse de notre méthode est quasiment aussi rapide qu'une méthode classique utilisant un calcul linéaire des déformations, sans gestion des grands déplacements.

nb noeuds	44	132	198	225	756	1386	3636	6666
nb tétraèdres	100	600	1000	2250	5000	10000	25000	50000
Hz notre méthode	300	75	50	30	12	5	3	1
Hz tenseur linéaire classique	340	82	55	32	13	5	3	1

(pentium 4 2.4GHz, 512 Mo de RAM, GeForce 4 MX)

TAB. 1 – Vitesse d'affichage en fonction du nombre d'éléments (contenant les calculs nécessaire au rendu)

5 Conclusion

Nous avons proposé dans cet article une nouvelle technique pour animer des solides déformables de façon rapide et robuste. Un nouveau calcul des déformations, basé sur un tenseur linéaire et des changements de repères, améliore la vitesse tout en gérant les grands déplacements, et une mise à jour de la matrice de rigidité permet d'assurer une grande stabilité même en grandes déformations. La robustesse de notre méthode s'appuie essentiellement sur le respect des principes de conservation des quantités de mouvement et de l'énergie cinétique. Notre approche permet des simulations interactives pour des modèles relativement détaillés. Dans le futur, nous avons l'intention d'utiliser la méthode sur d'autres types d'éléments, et particulièrement les hexaèdres. Il pourrait aussi être intéressant de compléter le simulateur avec des matériaux plus complexes et de gérer les changements de topologie. Et pour concentrer les ressources là où elles sont le plus nécessaires, nous désirons adopter une approche multirésolutions.

Références

- [BW98] D. Baraff and A. Witkin. Large steps in cloth simulation. In *Computer Graphics Proceedings, Annual Conference Series*. ACM Press / ACM SIGGRAPH, 1998. Proceedings of SIGGRAPH.
- [CDA00] S. Cotin, H. Delingette, and N. Ayache. A hybrid elastic model allowing real-time cutting, deformations and force-feedback for surgery training and simulation. *The Visual Computer*, 16(8) :437–452, 2000.
- [CDC⁺96] S. Cotin, H. Delingette, J.-M. Clement, V. Tasseti, J. Marescaux, and N. Ayache. Volumetric deformable models for simulation of laparoscopic surgery. In *Proceedings of the International Symposium on Computer and Communication Systems for Image Guided Diagnosis and Therapy, Computer Assisted Radiology (CAR'96)*, volume 1124 of *International Congress Series*. Elsevier, June 1996.
- [DCB01] G. Debunne, M. Desbrun M-P. Cani, and A. H. Barr. Dynamic real-time deformations using space and time adaptive sampling. In *Computer Graphics Proceedings, Annual Conference Series*. ACM Press / ACM SIGGRAPH, Aug 2001. Proceedings of SIGGRAPH'01.
- [DDBC99] G. Debunne, M. Desbrun, A. H. Barr, and M-P. Cani. Interactive multiresolution animation of deformable models. In *Eurographics Workshop on Computer Animation and Simulation*, Sep 1999.
- [DDCB00] G. Debunne, M. Desbrun, M-P. Cani, and A. H. Barr. Adaptive simulation of soft bodies in real-time. In *Computer Animation 2000, Philadelphia, USA*, pages 133–144, May 2000.
- [EK03] O. Etzmuß and M. Keckeisen. A Linearised Finite Element Model for Cloth Animation. Technical Report WSI-2003-2, Universität Tübingen, 2003.
- [Gar02] J. Garrigues. Initiation à la méthode des éléments finis. In *Ecole Supérieure de Mécanique de Marseille (ESM2) Courses*, 2002.
- [ITF04] G. Irving, J. Teran, and R. Fedkiw. Invertible finite elements for robust simulation of large deformation. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2004.
- [MDM⁺02] M. Müller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler. Stable real-time deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 49–54, 2002.
- [MG04] M. Müller and M. Gross. Interactive virtual materials. In *Proceedings of Graphics Interface*, 2004.
- [MTG04] M. Müller, M. Teschner, and M. Gross. Physically based simulation of objects represented by surface meshes. In *Proceedings of Computer Graphics International*, 2004.
- [PDA00] G. Picinbono, H. Delingette, and N. Ayache. Improving realism of a surgery simulator : Linear anisotropic elasticity, complex interactions and force extrapolation. Technical Report RR-4018, INRIA, 2000.
- [PDA03] G. Picinbono, H. Delingette, and N. Ayache. Non-linear anisotropic elasticity for real-time surgery simulation. *Graph. Models*, 65(5) :305–321, 2003.
- [TF88] D. Terzopoulos and K. Fleischer. Modeling inelastic deformation : viscoelasticity, plasticity, fracture. In *Computer Graphics Proceedings, Annual Conference Series*. ACM Press / ACM SIGGRAPH, 1988. Proceedings of SIGGRAPH.
- [TPBF87] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In *Computer Graphics Proceedings, Annual Conference Series*. ACM Press / ACM SIGGRAPH, 1987. Proceedings of SIGGRAPH.