Stabilisation par filtrage de méthodes d'intégration explicites

Alexandre Ahmad et Olivier Terraz et Samir Adly et Djamchid Ghazanfarpour

XLIM - Département Mathématiques-Informatique - Université de Limoges

Abstract

Mass spring systems have proved their efficiency for deformable behaviour simulation. Animation of such models highly depends on the numerical integration method used for solving ordinary differential equations. Two main formulations exist to solve these equations, namely explicit and implicit. Explicit methods exhibit poor stability properties (due to the Courant Frederick Levy condition, implying small time steps), contrary to implicit modelling which benefits from unconditional stability (larger time steps can be taken). Part of this stability is due to force filtering which is inherent to implicit formulation. Taking advantage of this, this paper presents an original post-processing velocity filtering algorithm for explicit schemes to enable larger time steps. Implementation of this method is straightforward for existing physically-based simulators. Results on all tested explicit methods show a computational time acceleration of about 20%. This directly comes from an increase of the time step in the same proportion, resulting from a decrease of eigenvalues (due to filtering). We also analyse frequencies and define a stability criterion for this post-processing filter. Applications to cloth and fish animations are presented.

Résumé

L'utilisation de maillages masses ressorts est très répandue pour simuler des modèles déformables. L'animation de tels modèles dépend de la méthode d'intégration numérique choisie pour résoudre les équations différentielles ordinaires engendrées par le système. Il existe deux familles de méthodes pour résoudre ces équations : explicites et implicites. Les méthodes de résolution explicites ont des propriétés de stabilité très contraintes par la condition de Courant Frederick Levy qui impose de petits pas de progression temporelle. À l'opposé, les méthodes implicites disposent d'une stabilité inconditionnelle (théorique) qui permet l'utilisation de grands pas de progression temporelle. Ce papier présente un algorithme de filtrage post traitement pour les schémas explicites qui autorise de plus grands pas. Les expérimentations effectuées sur différentes méthodes explicites montrent une réduction des temps de calculs de 20%. Cette diminution est directement liée à un agrandissement du pas de progression temporelle, engendré par une baisse des valeurs propres des vitesses. L'implémentation de cet algorithme est directe dans le cadre de simulateurs basés physiques existants. Nous proposons aussi la définition d'un critère de stabilité pour la méthode et présentons une analyse fréquentielle. Des applications sur des maillages réguliers tels que des tissus et irréguliers tels que des poissons sont exposées.

1. Introduction

L'utilisation de principes physiques en animation est devenue incontournable. Dans ce cadre, la physique a pour but d'assister les artistes dans la création de mouvements réalistes. Les systèmes masses ressorts sont très employés pour les simulations basées physique. Ils sont utilisés dans de nombreux domaines, tels que la simulation de déplacement de serpents, de vers [Mil88] et de poissons [TT94], mais aussi pour la dynamique des fluides [MP89], également pour l'animation de tissus [BW98, CK02] et de plus en plus pour la chirurgie médicale [MBT03]. Pour évaluer les changements de positions des particules, les équations du système masse ressort sont transformées en équations différentielles ordinaires (EDO).

Résoudre numériquement des EDOs n'est pas une tâche aisée. Les méthodes explicites sont les plus simples pour les résoudre mais comportent de mauvaises propriétés de stabilité : la convergence n'est assurée que si le pas d'intégration temporelle est petit, ce qui implique un nombre important d'itérations par image. Les schémas explicites sont donc en pratique peu utilisés. La formulation implicite est beaucoup plus efficace car il n'y a pas de restrictions sur le pas. Cette

formulation, bien que nécessitant la résolution d'un système linéaire, reste cependant plus rapide. Une partie de cette efficacité repose sur du fitrage qui est engendré par la résolution du système linéaire. Ce filtrage est appelé l'amortissement artificiel [Kas95, MDDB01].

Ce papier présente l'application d'un filtrage similaire sur des méthodes explicites. Tout en préservant la simplicité de ces schémas, le pas d'intégration temporelle est agrandi, diminuant ainsi le nombre d'itérations et donc les temps de calculs. Ce filtrage est indépendant de la méthode d'intégration numérique utilisée. Les expériences sur les schémas classiques montrent des accélérations de calculs d'animations de 20% en moyenne. Pour des maillages modestement rigides, les temps de calculs obtenus par notre approche et le modèle implicite sont proches. Malheureusement, le filtrage ajoute de l'amortissement. Par conséquent les animations sont plus "lissées", visible notamment sur nos animations de tissus exposées. Nous présentons aussi des expérimentations sur des maillages irréguliers avec des coefficients hétérogènes tel que le modèle de poisson proposé dans [TT94].

Nos comparaisons montrent que les animations calculées par des schémas explicites contiennent des hautes fréquences, tandis que les mêmes animations calculées avec une méthode implicite révèlent des fréquences plus basses. Une analyse du vecteur de vitesses indique que le filtrage réduit leur valeur propre maximale qui est directement responsable de la taille du pas d'intégration temporelle [HES03, Shi04]. Les résultats d'expérimentations montrent que l'algorithme de filtrage prend tout au plus 5% du temps de calcul.

Cet article est organisé comme suit : la section 2 présente un survol de l'état de l'art. La section 3 décrit le problème de la propagation des forces. Nous montrons comment retarder l'instabilité en filtrant dans la section 4. Les analyses et les comparaisons figurent dans la section 5, puis nous présentons nos conclusions et travaux futurs en section 6.

2. État de l'art

Le calcul du déplacement des systèmes masse ressort repose sur la méthode d'intégration numérique choisie. Une vue d'ensemble des schémas d'intégration mis en oeuvre en synthèse d'images figure dans [VMT01, HES03]. La stabilité et la précision sont les deux critères principaux pour adopter une méthode d'intégration. Comme indiqué dans [NMK*05], dans le domaine de l'animation basée physique la stabilité est souvent plus importante que la précision.

Les deux familles de méthodes d'intégration numériques, i.e. explicite et implicite peuvent être divisées en deux catégories : les méthodes à pas unique et à pas multiples. Les méthodes à pas multiples ont recours à au minimum deux valeurs précédemment calculées pour obtenir la valeur suivante. Elles ne sont pas par conséquent adaptées à l'animation car les anciennes valeurs sont inconsistantes dans le cadre d'interactions utilisateurs ou bien de collisions. À l'exception des BDF2 (méthode implicite à pas multiples, voir [HE01, CK02]), la plupart des méthodes utilisées sont des modèles à pas unique.

Les bonnes performances du modèle implicite ont été révélées par Baraff et Witkin [BW98] dans le domaine de l'animation de tissus (des schémas similaires étaient utilisés pour la dynamique des fluides dans [KM90] et aussi pour animer des modèles déformables dans [Fle87, TF88]). Depuis, les améliorations de ce modèle stable se sont principalement intéressées aux optimisations des temps de calculs. Pour accélérer la résolution du système linéaire, principalement composée de l'algorithme du gradient conjugué, Volino et Magnenat-Thalmann [VT00] ont proposé d'exploiter la propriété creuse de la matrice en utilisant une alternative à cette structure de données. Meyer *et al.* [MDDB01] ont suggéré de décomposer les forces en termes linéaires et non-linéaires et de précalculer la matrice inverse du système dans le but d'obtenir des performances en temps réel. Choi et Ko [CK02] ont montré une source d'instabilité, les pliures. Les auteurs ont proposé un modèle simulant ce phénomène qui offre des animations stables et des effets de pliures réalistes.

Les méthodes explicites étaient très convoitées avant [BW98]. Miller a adopté la méthode intitulée *forward* Euler pour animer des vers et des serpents [Mil88]. Les schémas Runge-Kutta, d'ordre 2 et plus, ont recours à des valeurs intermédiaires, engendrant une précision et une stabilité meilleure. L'intégration de Verlet, aussi connue sous d'autres appellations, est probablement la méthode la plus employée en animation basée physique (voir [BFA02, KANB03]). Tout en conservant la simplicité de la méthode de *forward* Euler, l'intégration de Verlet est plus stable.

Des travaux ont porté sur la fusion des deux grandes catégories IMplicite/EXplicite, i.e. IMEX, pour profiter des avantages de chacune. Tu et Terzopoulos [TT94] ont résolu le système linéaire explicitement dans l'espace et implicitement dans le temps. Eberhardt *et al.* [EEH00] et Bridson *et al.* [BMF03] ont éclaté les forces en termes linéaires et non linéaires, calculés par une méthode respectivement implicite et explicite. Boxerman et Ascher [BA04] ont défini un seuil d'instabilité pour les méthodes explicites. Lorsque celui-ci est détecté, le système est résolu par un procédé implicite. Les auteurs séparent la matrice du système linéaire en matrices plus petites, en utilisant un procédé de décomposition de graphe. Les calculs d'animations sont alors plus rapides

Hormis le fait que les recherches s'intéressent d'avantages aux approches implicites et IMEX, les méthodes explicites sont toujours d'actualité. Kačić-Alesić *et al.* [KANB03] expliquent leurs avantages pour la production cinématographique. Nos expérimentations ont révélé des temps de calculs concurrentiels entre des schémas explicites filtrés et des méthodes implicites pour des exemples avec une

valeur propre maximale importante (10^6). De plus, notre approche est plus rapide pour des systèmes raisonnablement rigides. En revanche, pour de très grands coefficients de rigidité (10^9), la résolution implicite est beaucoup plus efficace, bien qu'aucun de nos exemples n'ai suscité de telles valeurs.

3. Définition du problème et solution

Cette section définie un des problèmes rencontré par la résolution explicite d'EDOs, appliqués au système masse ressort, i.e. la propagation des forces (voir aussi [Kas95, MDDB01]). La figure 1 illustre le cas unidimensionnel d'une corde.



Figure 1: *Un système masse ressort 1D.*

Un ensemble de particules uniformément espacées $\{m_1,\ldots,m_5\}$, de masse m identique, sont connectées par des ressorts avec un coefficient de raideur $k>0$. À t_0 aucune force n'est appliquée sur le système, les ressorts sont à leur longueur de repos, i.e. le système est en équilibre statique. Si k est petit et qu'une force est exercée sur la particule m_1, alors m_1 devrait se déplacer librement et n'affecter que légèrement les autres particules. En revanche si la valeur de k est importante, du fait de la propagation des forces, la structure entière devrait se déplacer instantanément.

3.1. Système d'EDO

La mécanique des systèmes masse ressort peut être formulée comme une équation différentielle de la forme :

$$\mathbf{M}\times\ddot{x}=f(x,\dot{x}) \tag{1}$$

où $x,\dot{x},\ddot{x}$ représentent respectivement les vecteurs de position, de vitesse et d'accélération de taille $3n$ (dans un repère 3D), où n représente le nombre de particules, f est le vecteur de forces et $\mathbf{M}$ est une matrice diagonale contenant les masses (de taille $3n\times 3n$, i.e. $diag(\mathbf{M})=m_1,m_1,m_1,m_2,m_2,m_2,\ldots$). En faisant le changement de variable suivant, $v\equiv\dot{x}$, l'équation 1 peut être réécrite comme suit :

$$\frac{d}{dt}\begin{pmatrix} x\\ v\end{pmatrix}=\begin{pmatrix} v\\ \mathbf{M}^{-1}\times f(x,v)\end{pmatrix} \tag{2}$$

La section suivante analyse le comportement des schémas explicite et implicite appliqués à l'équation 2.

3.2. Un des problèmes de l'intégration explicite

La formule de la méthode *forward* Euler est la suivante :

$$\begin{pmatrix} x^{t+\Delta t}\\ v^{t+\Delta t}\end{pmatrix}=\begin{pmatrix} x^{t}\\ v^{t}\end{pmatrix}+\Delta t\begin{pmatrix} v^{t}\\ \mathbf{M}^{-1}\times f(x^t,v^t)\end{pmatrix} \tag{3}$$

Connaissant les conditions initiales, l'équation 3 est résolue itérativement. Les ressorts agissent uniquement sur les deux particules connectées aux extrémités. Dans cette équation v^t est utilisée pour mettre à jour $x^{t+\Delta t}$ et non x^t (une extension simple connue sous le nom de *forward-backward* (FB) Euler tire profit de $v^{t+\Delta t}$). Par conséquent un déplacement de la particule m_1 affectera m_2 après deux itérations. Les forces agissant sur m_1 sont donc propagées sur m_5 après $2\times n$ itérations. Il en résulte que les particules ont une certaine liberté de mouvement sans effet immédiat sur l'ensemble de la structure, i.e. des variations locales. Il va de soi que la propagation des forces devrait être instantanée, ce qui est le cas avec la méthode de *forward* Euler qu'après $2\times n^2$ itérations par pas d'intégration et n'est donc pas envisageable en pratique. La formulation implicite solutionne cet inconvénient.

3.3. La solution implicite

La méthode intitulée *backward* Euler est la plus connue des formulations implicites. Elle est définie comme suit :

$$\begin{pmatrix} x^{t+\Delta t}\\ v^{t+\Delta t}\end{pmatrix}=\begin{pmatrix} x^{t}\\ v^{t}\end{pmatrix}+\Delta t\begin{pmatrix} v^{t+\Delta t}\\ \mathbf{M}^{-1}\times f(x^{t+\Delta t},v^{t+\Delta t})\end{pmatrix} \tag{4}$$

Ce procédé implique le calcul approximatif de $f(x^{t+\Delta t},v^{t+\Delta t})$ en linéarisant (voir [BW98]). La deuxième ligne de l'équation 4 peut être mise sous la forme du système linéaire suivant :

$$\mathbf{A}v=b \tag{5}$$

où $\mathbf{A}$ est une matrice, souvent creuse, symétrique et définie positive (ou transformée pour l'être), v est le vecteur de vitesse cherché et b est un vecteur temporaire. Il existe différentes méthodes pour résoudre les systèmes linéaires, voir dans [PTVF92]. Pour en nommer quelques unes, la factorisation de Cholesky et la méthode du gradient conjugué sont les plus utilisées en animation, avec une grande préférence pour cette dernière due à sa vitesse de résolution (voir [BW98, CK02, BA04]). Nous nous intéressons à la factorisation de Cholesky, où la matrice $\mathbf{A}$ est décomposée en deux matrices triangulaires, i.e. une inférieure $\mathbf{L}$ et sa transposée $\mathbf{L^T}$.

$$\mathbf{A}=\mathbf{L}\mathbf{L}^{\mathbf{T}} \tag{6}$$

En appliquant cette factorisation au système illustré par la figure 1, $\mathbf{A}$ et $\mathbf{L}$ ont les formes suivantes :

$$\mathbf{A}=\begin{pmatrix} a_{0,0} & a_{1,0} & & & \mathbf{0}\\ a_{1,0} & a_{1,1} & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & a_{4,4} & a_{4,5}\\ \mathbf{0} & & & a_{5,4} & a_{5,5}\end{pmatrix}\quad \mathbf{L}=\begin{pmatrix} l_{0,0} & & & & \mathbf{0}\\ l_{1,0} & l_{1,1} & & & \\ & l_{2,1} & \ddots & & \\ & & \ddots & l_{4,4} & \\ \mathbf{0} & & & l_{5,4} & l_{5,5}\end{pmatrix}$$

La solution v est obtenue par une substitution *forward* (équation 7) et *backward* (équation 8).

$$\mathbf{L}y = b \tag{7}$$

$$\mathbf{L}^{\mathbf{T}}v = y \tag{8}$$

L'analyse de ce calcul permet de comprendre le procédé mis en oeuvre par les méthodes implicites pour gérer la propagation des forces. En considérant l'équation 7, appliquée à notre exemple, le premier élément est simple à calculer, i.e. $y_0 = b_0/l_{0,0}$. Les autres éléments sont calculés avec la récurrence suivante : $y_i = \left(b_i - y_{i-1} \times l_{i,i-1}\right)/l_{i,i}$. Comme indiqué dans [Kas95], le cas où $l_{i,i-1} = 1/a$ et $l_{i,i} = (a-1)/a$ représente des valeurs usuelles en dehors des extrémités. Cette récurrence peut être récrite sous la forme suivante :

$$y_i \;=\; \lambda \times b_i + (1-\lambda) \times y_{i-1} \tag{9}$$

où $\lambda = a \times (a-1)^{-1}$. L'équation 9 est un filtre récursif. La valeur de sortie y_i est une pondération entre la valeur précédente y_{i-1} et la valeur courante b_i. C'est donc un filtre passe bas avec un noyau de lissage déterminé par λ. Si les particules adjacentes ont des variations de force similaires, ce qui est le cas lorsque le pas d'intégration est petit, alors le filtrage a un effet limité. En revanche, le filtrage prend tout son sens lorsque les forces sont variées (ce qui est le cas lorsque le pas est grand).

Un filtrage passe bas est imbriqué dans la résolution du système linéaire. En d'autres termes les particules propagent leurs forces à leurs voisines, aux voisines des voisines, etc., et ce, durant un unique pas d'intégration. Plus le pas est grand, plus le filtrage est important et inversement. Par conséquent la propagation des forces n'est plus un problème de stabilité. En prenant avantage de ce procédé, nous proposons d'effectuer un filtrage identique sur des systèmes masses ressorts dont le déplacement est calculé par des méthodes explicites, afin d'autoriser des pas d'intégration temporelle plus importants.

4. Notre solution pour l'intégration explicite

La section précédente explique le filtrage passe bas engendrée par la résolution implicite. Ainsi les variations locales, i.e. les hautes fréquences, sont atténuées et donc la stabilité est améliorée. Pour propager des forces avec des méthodes explicites, nous proposons de diffuser la vélocité des particules à ses voisines. Nos expérimentations nous ont montré que le filtrage des forces offre des résultats similaires, ce qui n'est pas surprenant puisque les vitesses sont directement liées aux forces. Filtrer les vélocités permet d'appliquer cet algorithme en post traitement, sans avoir à modifier le modèle d'intégration existant. La méthode d'intégration numérique est donc indépendante. Les résultats indiquent qu'en fonction du noyau de lissage du filtre, des pas plus important peuvent être employés (voir la figure 3).

L'idée est de calculer la vélocité d'une particule en fonction d'une pondération de ses voisines. Pour ce faire, la nouvelle vitesse est en premier lieu calculée par une méthode d'intégration numérique quelconque (comme une boite noire). En second lieu, la vélocité est modifiée en fonction des vitesses voisines (voir l'algorithme 1). En dernier lieu les positions sont mises à jour.

4.1. Structure du modèle

Pour nos expérimentations, nous avons appliqué notre méthode sur différentes structures de masses ressorts, telles que des tissus, des cubes et des poissons (ce modèle est particulièrement intéressant puisque les coefficients de rigidité, d'amortissement et de masse sont différents tout au long de l'objet). Dans un souci de clarté, toutes les expériences illustrées concernent les tissus (2D). La section 5 présente nos résultats sur les autres modèles. Nous avons utilisé la structure proposée dans [CK02].

4.2. Filtrage proposé

L'équation 9 montre un filtre exponentiel généré par une résolution implicite. Nous avons appliqué un filtrage identique, i.e. exponentiel. Nous avons aussi expérimenté un filtrage Gaussien et avons obtenu des résultats similaires, i.e. des augmentations du pas d'intégration dans les même proportions et un amortissement du mouvement visuellement comparable. Le filtre exponentiel est défini par :

$$\beta(d) = \lambda \times e^{-\lambda|d|} \tag{10}$$

où d est la distance entre deux particules. La figure 2 illustre le paramétrage du filtre en fonction du coefficient λ. Un filtrage adaptatif peut donc être exercé en variant ce coefficient. Il est important de noter qu'un filtrage excessif, i.e. lorsque $\lambda < 1$, engendre des effets indésirables car le mouvement est trop amorti. Dans ce cas les vitesses des particules voisines ont trop d'influence et réduisent considérablement l'action de la particule filtrée. Il n'y a plus de variations locales, ne subsiste qu'un mouvement global, parfois même statique. En

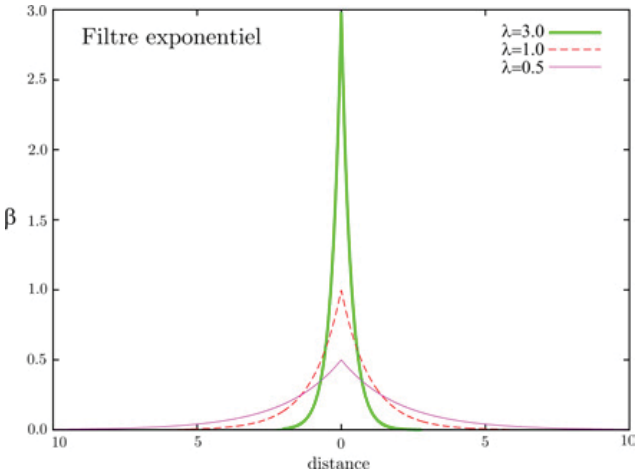


Figure 2: *Le filtre exponentiel*

considérant la i-ème particule, sa nouvelle vitesse filtrée est calculée comme suit:

$$v_{i_{filtre}}^{t+\Delta t} = \left(\lambda \times v_i^{t+\Delta t} + \sum_{j \in voisins(i)} \beta_j^{t+\Delta t} \times v_j^{t+\Delta t}\right)$$

où $\beta_j^{t+\Delta t}$ est la pondération des valeurs voisines. Les coefficients $\beta_j^{t+\Delta t}$ varient au cours du temps et doivent être normés. Cependant l'échange entre deux particules i et j n'est pas symétrique et peut produire des effets indésirables. Nous proposons la simple alternative suivante :

$$v_{i_{filtre}}^{t+\Delta t} = \left(\lambda \times v_i^{t+\Delta t} + \sum_{j \in voisins(i)} \beta_j^{t+\Delta t} \times \left(v_j^{t+\Delta t} + v_i^{t+\Delta t} \right) \right)$$

Cette procédure est détaillée dans l'algorithme 1.

L'algorithme de notre méthode est semblable à celui proposé dans [MDDB01]. Cependant, il y a des différences notables. Les auteurs proposent de simplifier la matrice **A** du système linéaire. Une matrice inverse est précalculée et sert de filtrage pour les forces. Leur but est d'accélérer la résolution implicite. Nous proposons un filtre générique pour les méthodes explicites qui permet d'agrandir le pas d'intégration temporelle et donc permet de diminuer les temps de calculs. De plus nous présentons une analyse fréquentielle.

Algorithm 1 L'algorithme de filtrage

Require: vecteur vel, vecteur pos, λ
1: initialisation du vecteur 3D v^{filtre}
2: **for all** particule i **do**
3: $norme \leftarrow \lambda$
4: $v_i^{filtre} \leftarrow \lambda \times v_i$ {la particule courante}
5: **for all** particule $j \in voisins(i)$ **do**
6: $d \leftarrow dist(x_i, x_j)$ {calcul des distances}
7: $\beta = \lambda \times e^{-\lambda |d|}$ {calcul du coefficient}
8: $v_i^{filtre} \leftarrow v_i^{filtre} + \beta \times (v_j + v_i)$ {filtrage}
9: $norme \leftarrow norme + \beta$
10: **end for**
11: $v_i^{filtre} \leftarrow v_i^{filtre} / (2 \times norme)$ {mise à l'échelle}
12: **end for**
13: **return** v^{filtre}

4.3. Implémentation

L'implémentation de notre proposition est simple à mettre en oeuvre pour des calculateurs d'EDOs existants. Notre approche s'est montrée opérationnelle pour toutes les méthodes explicites testées, i.e. les méthodes de *forward* Euler, FB Euler, Verlet et Runge-Kutta 4 (RK4). Dans nos exemples nous avons utilisé les formules de calculs des forces de friction (avec le fluide environnant), de rigidité et d'amortissement (des ressorts) comme définies dans [TT94].

4.4. Taille du pas d'intégration

Pour déterminer le pas d'intégration, nous effectuons une analyse des valeurs propres des vitesses. Pour les méthodes de *forward* Euler, FB Euler et Verlet, le pas maximal autorisé peut être calculé comme suit:

$$\Delta t_{max} = \frac{2}{\sqrt{k_0}} \tag{11}$$

où k_0 est la valeur propre maximale calculée à partir de la matrice de rigidité (voir [HES03, Shi04]). Une méthode simple et efficace pour déterminer k_0 est la méthode des puissances. Pour nos besoins nous avons modifié cet algorithme en intégrant le filtrage (voir l'algorithme 2). Les résultats ont confirmé nos prédictions (voir la figure 3). Comme

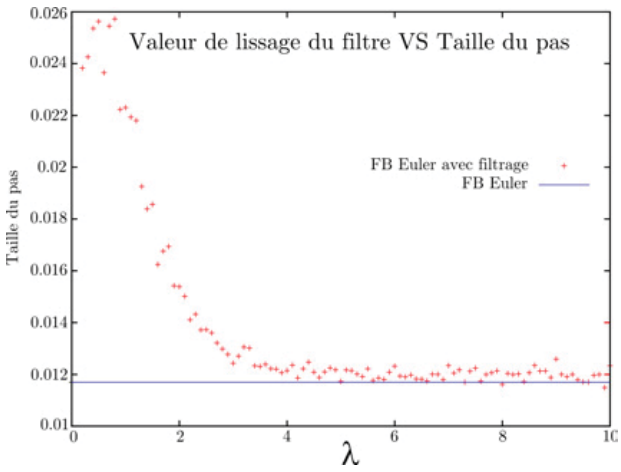


Figure 3: *La taille du pas d'intégration temporelle est déterminée par* λ.

dans la formulation implicite, plus le pas d'intégration est grand, plus le filtrage est important. Lorsque $\lambda = 1$, le pas d'intégration est doublé. Les animations sont alors calculées quasiment deux fois plus rapidement. Dans ce cas k_0 est divisé par quatre (voir l'équation 11). Si les coefficients des ressorts et des masses ne changent pas pendant la durée de l'animation (ce qui est souvent le cas), alors k_0 est du même ordre, ce qui implique un pas d'intégration temporelle maximal constant. Cet algorithme est donc exécuté une seule fois pour toute, en guise de précalcul. Puisque peu d'itérations sont suffisantes pour que l'algorithme 2 converge, alors sa complexité est de l'ordre de $\mathcal{O}(n)$ (en utilisant la structure de données proposée par [VT00]). Certains de nos exemples ont supporté sans inconvénients cette augmentation double du pas, ce qui a pour effet de réduire les temps

Algorithm 2 La méthode des puissances modifiée

Require: vecteur pos, lambda
1: v vecteur (aléatoire)
2: **for** $k = 1, 2, 3, \ldots$ jusqu'à la convergence **do**
3: $v \leftarrow A \cdot v$
4: $\alpha \leftarrow max(v)$
5: $v \leftarrow v / \alpha$
6: $v \leftarrow filtrer(pos, v, \lambda)$
7: **end for**
8: **return** α

de calculs d'animations de presque 50%. D'autres ne l'ont pas permis, car la propagation des forces n'est pas la seule cause d'instabilité des méthodes explicites. Dans le cas où une explosion numérique est détectée, le filtrage atténue la vitesse. Il peut corriger la vélocité erronée d'une particule en fonction des vitesses des particules voisines (tout dépend de l'erreur bien sûr). En revanche si toutes les vitesses sont erronées, alors le filtrage ne peut plus rien atténuer et ne peut empêcher une explosion numérique. C'est un cas typique des système rigides. Tous nos exemples ont cependant autorisé une augmentation de 25% du pas. Il est à noter qu'avec cette valeur sans filtrage, le système explose (voir la vidéo "FB Euler VS FFB Euler"). Le nombre de voisins est constant et est usuellement beaucoup plus petit que le nombre de particules n, donc la complexité de l'algorithme de filtrage est en $\mathcal{O}(n)$. Nous avons expérimenté des temps calculs d'animations 20% plus court pour des pas d'intégration 25% plus grands (ce qui implique que le filtrage a un coût de 5%).

5. Résultats

Pour toutes nos expérimentations, la plateforme de test utilisée est un AMD 2800+ sous Linux. Les vidéos sont disponibles à l'adresse suivante : http://www.msi.unilim.fr/∼ahmad/afig2006/.

5.1. Analyses des vitesses et fréquences

Les animations calculées par des méthodes explicites, du fait des variations locales plus importantes, devraient être plus riches en détails, tandis que les mouvements calculés par une approche implicite ou explicite filtrée sont supposés être plus lissés, ou amortis.

Pour analyser les vitesses, nous avons décidé de représenter le champ de vélocité dans un espace RGB au lieu d'un espace 3D. Nous avons effectué l'expérience suivante : un maillage 2D, attaché en deux extrémités, est soumis aux forces de gravité pendant 10 secondes (voir la figure 5). Trois simulations sont testées avec respectivement les méthodes suivantes (voir la vidéo "expérimentations") : FB Euler, *backward* Euler et FB Euler filtrée (FFB Euler). La figure 6

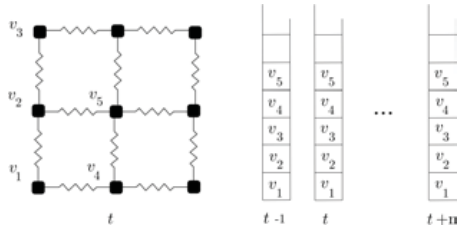


Figure 4: *À chaque pas d'intégration, les vitesses sont stockées dans un tableau unidimensionnel.*

illustre l'évolution des vitesses tout au long des expérimentations. Aucun algorithme de pas adaptatif n'est utilisé, tous les coefficients sont identiques pour les trois simulations. L'axe horizontal représente le temps avec un pas de 0.01s. La figure 4 montre la transformation d'un maillage 2D en tableau 1D.

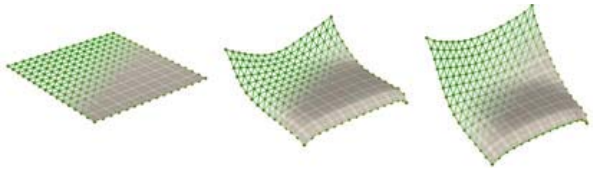


Figure 5: *un tissu est soumis à la gravité, attaché en deux extrémités.*

La figure 6 révèle des blocs de couleurs similaires qui existent dans les trois images et coïncident temporellement. La figure du haut contient beaucoup de perturbations (calculée avec la méthode FB Euler). Elles indiquent la présence de variations locales, i.e. des hautes fréquences. Comme prédit, la figure du milieu, calculée par *backward* Euler, contient peu de perturbations, puisqu'elles sont automatiquement filtrées. Nous avons résolu le système linéaire en utilisant la décomposition de Cholesky et la méthode du gradient conjugué. Dans les deux cas un filtrage est généré. La figure du bas est obtenue par la méthode FFB Euler. L'animation résultante est visuellement proche de celle calculée par la formulation implicite, notre modèle répond donc à nos attentes.

La figure 4 représente un maillage rectangulaire, représentant un ensemble de vitesses à l'instant t. Afin de visualiser le domaine fréquentiel, nous avons réalisé une transformée de Fourier 2D sur les vitesses du maillage tout en effectuant les même expérimentations que précédemment (voir les vidéos "analyses des fréquences"). Ces résultats sont visibles dans la figure 7. La valeur au centre des images de la ligne du bas, i.e. le domaine fréquentiel, indique la présence de basses fréquences. Les valeurs éloignées du centre correspondent aux hautes fréquences. Encore une fois les résultats répondent aux attentes : l'image calculée avec la méthode de FB Euler (à gauche) contient des fréquences plus élevées que les deux autres. L'illustration du milieu (schéma implicite) comporte des fréquences plus basses (près du centre). Des résultats visuellement similaires sont obtenus en filtrant la méthode explicite FB Euler (droite).

5.2. Comparaisons des temps de calculs

Afin d'illustrer l'efficacité de notre approche, nous avons fait varier k_0 (dû à la rigidité des ressorts) et le nombre de particules (les méthodes numériques sont généralement dépen-

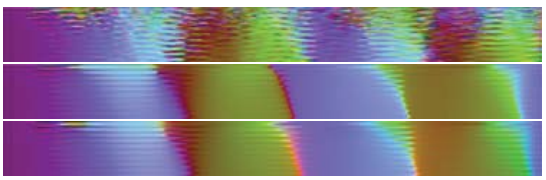


Figure 6: *L'évolution des vitesses calculées par :* haut: *FB Euler* milieu: *backward Euler* bas: *FFB Euler*

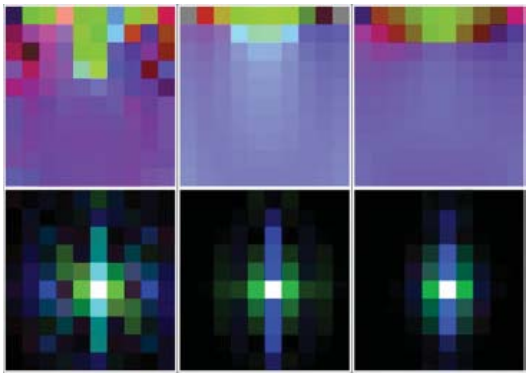


Figure 7: En haut *: un maillage de tissu 2D projeté dans une image.* En bas *: leur domaine fréquentiel correspondant.* De gauche à droite *: FB, backward et FFB Euler.*

dantes de ces deux paramètres). Nous avons expérimenté avec des méthodes explicites (FB Euler, RK4, et Verlet), une méthode implicite (*backward* Euler décrite dans [BW98]) et une méthode IMEX proposée dans [EEH00]. Pour des comparaisons justes, nous avons utilisé la structure de données s'affranchissant de matrices proposée dans [VT00] pour les méthodes IMEX et implicite. Pour ces deux modèles, le pas d'intégration temporelle est fixée à 0.01s. Pour les méthodes explicites nous précalculons le pas d'intégration maximal (cependant toujours inférieur ou égal à 0.01s) en utilisant l'équation 11 pour les méthodes de FB Euler et de Verlet et nous avons déterminé le pas pour le schéma de RK4 par cette équation : $\sqrt{8.75}/\sqrt{k_0}$ (voir [Shi04]). Les collisions sont mises hors d'usage pour des comparaisons justes, puisqu'elles sont en général gérées par des procédés différents. La figure 8 montre des comparaisons de temps de calculs en fonction du nombre de particule (avec k_0 fixé à 10^6).

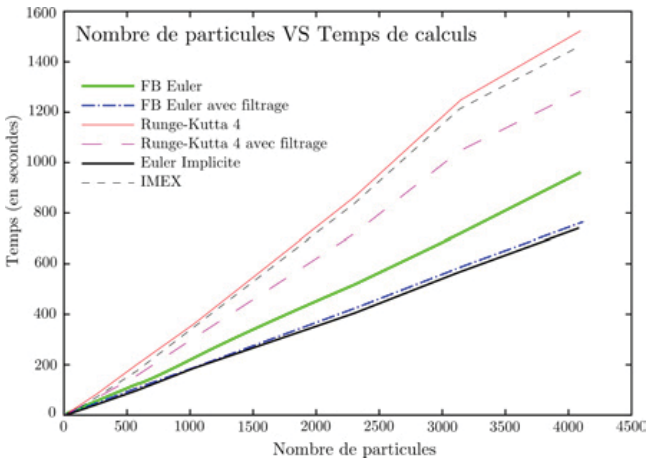


Figure 8: *Comparaisons entre les temps de calculs et le nombre de particules, pour* $k_0 = 10^6$.

Les résultats révèlent une accélération des temps de calculs d'environ 20% quand les schémas explicites sont filtrés (avec un pas d'intégration 25% plus grand). La méthode de Verlet n'est pas montrée pour des raisons de clarté mais est sujette à des améliorations équivalentes. Il est important de noter que l'algorithme du gradient conjugué (qui occupe majoritairement le calcul des méthodes implicites) dépend grandement du critère de convergence (le seuil d'erreur ε et le nombre d'itération maximum). Dans notre cas nous avons utilisé un seuil relativement grand, i.e. $\varepsilon = 0.01$ (stable cependant). Mais lorsque ε est réduit, la résolution est alors plus longue et la méthode de FFB Euler est alors plus rapide. La figure 9 montre des comparaisons de temps de calculs lorsque k_0 est augmenté. Les méthodes implicites et IMEX deviennent incontestablement plus efficaces lorsque $k_0 > 10^7$. Cependant avec de tels coefficients de raideurs les animations montrent des comportements de solides quasiment non déformables. Dans ce cas les méthodes explicites sont à éviter.

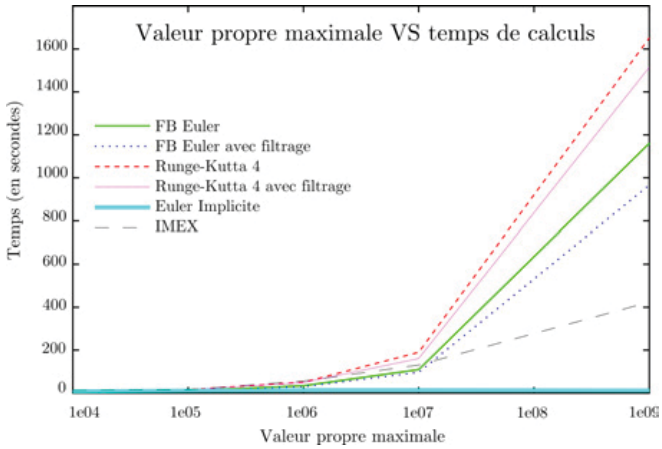


Figure 9: *Les méthodes explicites ont des temps de calculs compétitifs jusqu'à ce que* $k_0 = 10^7$.

L'application de notre méthode sur le maillage de poisson proposé dans [TT94] a révélé des résultats du même ordre, i.e. une augmentation possible du pas d'intégration de 25%. Les simulations calculées par les trois méthodes, i.e., explicite, implicite et explicite filtrée sont visuellement similaires. Il existe cependant une légère différence dans la propulsion uniquement quand le pas est inférieur à 0.01 (voir la vidéo "poisson").

6. Conclusions et travaux futurs

Les méthodes explicites disposent de propriétés intéressantes, telles que la simplicité et la précision. Malheureusement elles sont instables et nécessitent des pas d'intégration temporelle petit. Pour des systèmes rigides, les temps de calculs sont alors importants. Nous avons montré la mise en place d'un filtrage automatique des forces dans la formulation implicite, ce qui permet d'éviter certains problèmes de stabilité. En tirant profit de ce procédé, nous avons appliqué un filtrage similaire sur des schémas explicites. Les résultats et les analyses montrent que les pas d'intégration temporelle peuvent être agrandis de 25% jusqu'à 100%, et par conséquent les temps de calculs sont écourtés de 20%

à presque 50%. L'inconvénient est l'ajout d'un amortissement, les animations sont donc plus "lissées" (en fonction du coefficient de filtrage). Pour des exemples concrets, ($k_0 \leq 10^6$), notre méthode révèle des temps compétitifs comparés à ceux d'une formulation implicite. Mais les méthodes explicites sont inutilisables pour des exemples avec des coefficients de rigidité plus importants, même en prenant en compte notre accélération. Pour résoudre ce problème, nous proposons d'appliquer d'autres procédés issus des formulations implicites tels que la linéarisation. Nous planifions aussi d'améliorer le filtrage, en optant pour des filtres plus puissants tel que celui de Kalman, récemment mis en oeuvre pour des travaux connexes [GM06]. Une analyse de Von Neumann permettrait de quantifier les fréquences et de couper uniquement les indésirables. Pour des performances meilleures, notre algorithme pourrait être exécuté sur GPU.

References

[BA04] BOXERMAN E., ASCHER U.: Decomposing cloth. In *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2004), ACM Press, pp. 153–161.

[BFA02] BRIDSON R., FEDKIW R., ANDERSON J.: Robust treatment of collisions, contact and friction for cloth animation. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 594–603.

[BMF03] BRIDSON R., MARINO S., FEDKIW R.: Simulation of clothing with folds and wrinkles. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), Eurographics Association, pp. 28–36.

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1998), ACM Press, pp. 43–54.

[CK02] CHOI K.-J., KO H.-S.: Stable but responsive cloth. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 604–611.

[EEH00] EBERHARDT B., ETZMUSS O., HAUTH M.: Implicit-explicit schemes for fast animation with particle systems. In *Eurographics Computer Animation and Simulation Workshop 2000* (2000).

[Fle87] FLEISHER D. T. . J. P. . A. B. . K.: Elastically deformable models. *Computer Graphics 21*, 4 (July 1987), 205–214.

[GM06] GROTE M. J., MAJDA A. J.: Stable time filtering of strongly unstable spatially extended systems. *Proceedings of the National Academy of Sciences of the United States of America 103*, 20 (2006), 7548–7553.

[HE01] HAUTH M., ETZMUSS O.: A high performance solver for the animation of deformable objects using advanced numerical methods. In *Proc. Eurographics 2001* (2001), Chalmers A., Rhyne T.-M., (Eds.), vol. 20(3) of *Computer Graphics Forum*, pp. 319–328.

[HES03] HAUTH M., ETZMUSS O., STRASSER W.: Analysis of numerical methods for the simulation of deformable models. *The Visual Computer 19*, 7-8 (2003), 581–600.

[KANB03] KAČIĆ-ALESIĆ Z., NORDENSTAM M., BULLOCK D.: A practical dynamics system. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), Eurographics Association, pp. 7–16.

[Kas95] KASS M.: An introduction to physically based modeling, chapter introduction to continuum dynamics for computer graphics. In *SIGGRAPH Course Notes* (1995).

[KM90] KASS M., MILLER G.: Rapid, stable fluid dynamics for computer graphics. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1990), ACM Press, pp. 49–57.

[MBT03] MACIEL A., BOULIC R., THALMANN D.: Deformable tissue parameterized by properties of real biological tissue. In *IS4TH* (2003), pp. 74–87.

[MDDB01] MEYER M., DEBUNNE G., DESBRUN M., BARR A. H.: Interactive animation of cloth-like objects in virtual reality. *Journal of Visualization and Computer Animation 12*, 1 (2001), 1–12.

[Mil88] MILLER G. S. P.: The motion dynamics of snakes and worms. In *SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1988), ACM Press, pp. 169–173.

[MP89] MILLER G. S. P., PEARCE A.: Globular dynamics: A connected particle system for animating viscous fluids. *Computers & Graphics 13*, 3 (1989), 305–309.

[NMK*05] NEALEN A., MÜLLER M., KEISER R., BOXERMANN E., CARLSON M.: Physically based deformable models in computer graphics. In *Proceedings of Eurographics 2005* (2005), pp. 91–94.

[PTVF92] PRESS W. H., TEUKOLSKY S. A., VETTERLING W. T., FLANNERY B. P.: *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 1992.

[Shi04] SHINYA M.: Stabilizing explicit methods in spring-mass simulation. In *Computer Graphics International* (2004), pp. 528–531.

[TF88] TERZOPOULOS D., FLEISCHER K.: Modeling inelastic deformation: viscolelasticity, plasticity, fracture. In *SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1988), ACM Press, pp. 269–278.

[TT94] TU X., TERZOPOULOS D.: Artificial fishes: physics, locomotion, perception, behavior. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1994), ACM Press, pp. 43–50.

[VMT01] VOLINO P., MAGNENAT-THALMANN N.: Comparing efficiency of integration methods for cloth simulation. In *Computer Graphics International* (2001), pp. 265–274.

[VT00] VOLINO P., THALMANN N. M.: Implementing fast cloth simulation with collision response. In *CGI '00: Proceedings of the International Conference on Computer Graphics* (Washington, DC, USA, 2000), IEEE Computer Society, p. 257.