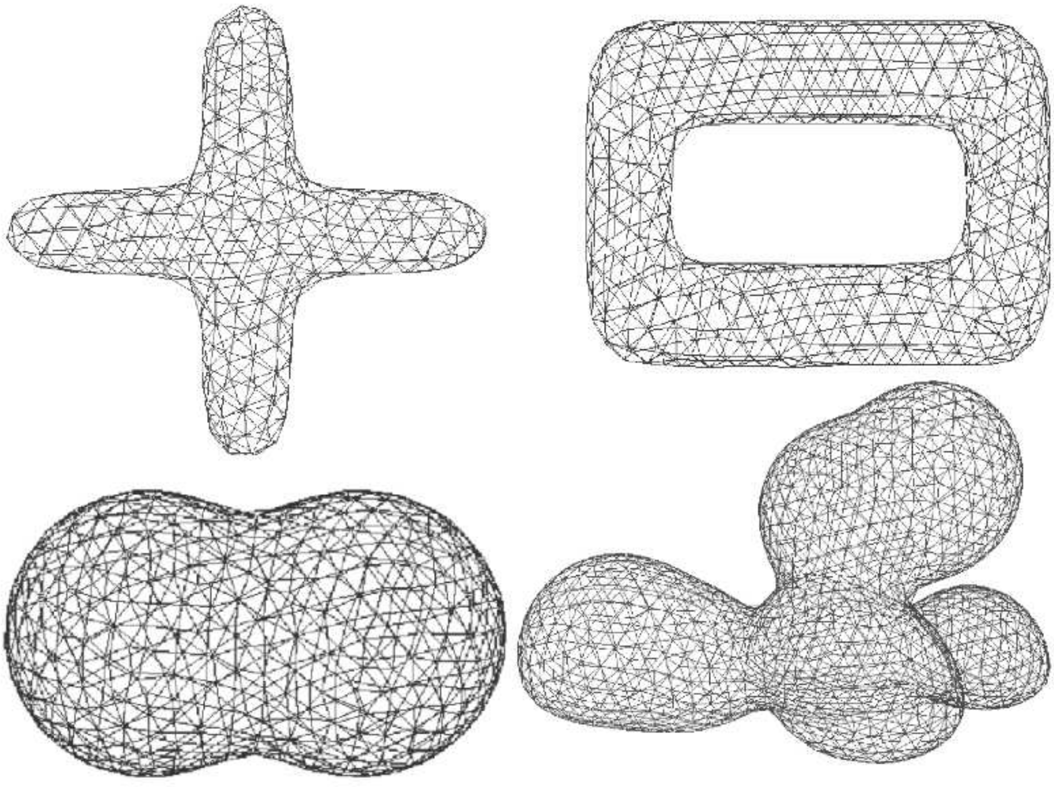
Triangulation de surfaces implicites dynamiques.

Benoît Laurent



Groupe ESIEE
Stage de I4
Majeure Informatique

IMAG-INRIA
Laboratoire Gravir
Projet Evasion

année 2004 - 2005

Table des matières

Remerciements

Je tiens à remercier tout d'abord Marie-Paule Cani, la responsable du projet Évasion, pour m'avoir accueilli dans son équipe, Antoine Bouthors pour m'avoir proposé ce stage très intéréssant et pour son suivi tout au long de ces quatre mois. Je remercie également l'équipe ÉVASION dans son ensemble, pour l'accueil qu'on m'a réservé, avec une mention spéciale pour Franck Hetroy et ses lumières en géométrie ainsi que Florence Bertails pour m'avoir supporté dans son bureau.

Présentation du stage

Au cours de cette année de I4 majeure informatique j'ai été amené à faire un stage de quatre mois dans un laboratoire d'informatique à Grenoble, hébergé au sein des bâtiments de l'INRIA Rhônes-Alpes. Ce stage m'a été proposé par Antoine Bouthors, ancien élève de l'ESIEE, qui effectue maintenant une thèse sur la simulation temps réel de nuages convectifs de type cumulus. Pour situer davantage les orientations de cette équipe voici une petite description, extraite de la page internet du laboratoire. *L'équipe EVASION (Environnements Virtuels pour l'Animation et la Synthèse d'Images d'Objets Naturels) du laboratoire GRAVIR (CNRS, INPG, INRIA, UJF) a été créée au* 1^er^ *janvier 2003. Elle regroupe cinq chercheurs ou enseignants-chercheurs permanents, onze étudiants en thèse et un ingénieur expert. Ses travaux de recherche sont dédiés à la modélisation, à l'animation, et à la visualisation d'objets et de phénomènes naturels. Pour cela, deux grands axes de recherche sont privilégiés : D'une part le développement d'outils fondamentaux destinés à la spécification de scènes et objets naturels complexes, à la mise au point de modèles alternatifs pour la forme, le mouvement et l'apparence ainsi qu'à la conception d'algorithmes reposant sur un niveau de détail adaptatif pour gérer au mieux la complexité ; d'autre part la validation de ces outils sur des scènes naturelles spécifiques, qui vont du monde minéral (océan, ruisseaux, lave, avalanches, nuages) au monde animal (simulation d'organes, visages corps et chevelure d'un personnage, mouvements d'animaux), en passant par les scènes végétales (morphogénèse de plantes, prairies, arbres).*

La thèse d'Antoine Bouthors inclue la simulation de nuages et le rendu réaliste en temps réel. Ce stage intervenait en amont de la phase de rendu, et devait permettre de mettre en œuvre une méthode de triangulation capable de s'adapter dynamiquement aux déformations des nuages. Cette triangulation servira de support pour les textures qui viendront donner aux nuages leur aspect réaliste.

Introduction

Compte tenu du sujet de ce stage j'ai été amené à utiliser de nombreuses notions, nouvelles et parfois originales, je commencerai donc par une courte description des outils utilisés par la suite. Suite à cette introduction, je détaillerai l'article [cRM96] qui décrit la méthode que j'ai mise en œuvre au cours de ce stage. Enfin je détaillerai les résultats obtenus, et les possibilités qu'ils nous offrent, pour la modélisation de nuages mais aussi pour d'autres domaines. Pour conclure je décrirai rapidement les autres méthodes envisageables, mais aussi les améliorations possibles de la méthode courante.

1 Présentation des outils mathématiques

1.1 Les surfaces implicites

Si l'on observe un cumulus avec plus d'attention on peut remarquer que sa forme globale est assez homogène et compacte. Cette caractéristique importante permet d'envisager la possibilité de modéliser un nuage du moins dans sa forme, de manière simple, par un ensemble de sphères agglutinées les unes aux autres, certaines aplaties à leur base[1] comme par exemple dans l'image suivante.

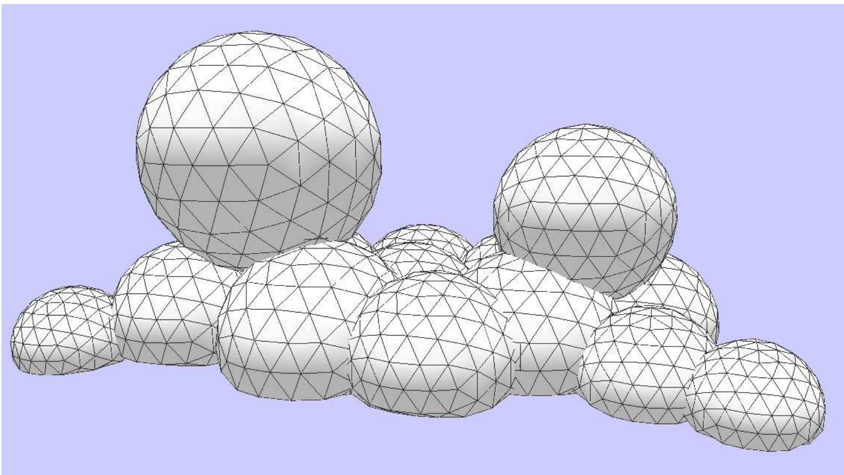


Fig. 1 – Exemple de nuage décrit à l'aide d'une surface implicite

Cette représentation très figurative parait très simpliste, elle donne pourtant une bonne approximation de la géométrie d'ensemble d'un nuage. De plus, il faut savoir que par dessus cette géométrie vient s'ajouter une texture qui donnera au nuage son aspect cotonneux, ainsi que son comportement à la lumière, i.e la transparence aux bords, le halo s'il est éclairé de derrière. Par exemple dans l'image suivante on voit bien que la bordure du nuage est quasiment transparente et qu'il y a une grande différence d'éclairage entre le bord et le centre du nuage. Le rendu de ces phénomènes a déjà été abordé durant son DEA par Antoine Bouthors [Bou04].

[1]Cette remarque n'est vraie que pour les cumulus.

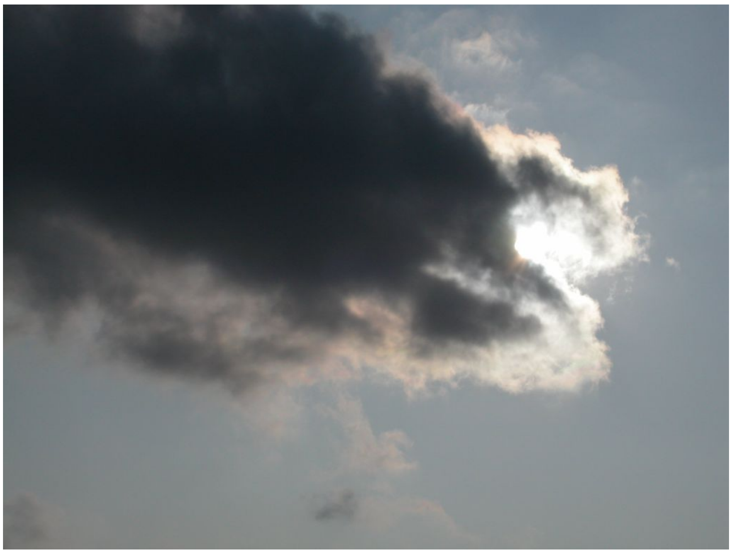


FIG. 2 – Exemple d'illumination d'un nuage naturel

Parmi les différentes représentations mathématiques de surfaces, il en est une qui convient parfaitement à l'utilisation qui nous intéresse : les surfaces implicites ou isosurfaces. En effet, elles sont particulièrement adaptées pour modéliser des surfaces très lisses et déformables. Si l'on se place dans le plan, les courbes de niveaux que l'on observe sur une carte IGN sont comparables à des courbes implicites puisque l'altitude est la même en chacun des points d'une courbe. De même, si l'on se place dans l'espace à trois dimensions on peut définir une fonction de potentiel à valeur dans $\mathbb{R}$, ainsi une isosurface sera définie par la surface passant par l'ensemble des points de l'espace de même potentiel. Par exemple : $f(r) = \frac{1}{r^2}$ avec $r = ||xc||$, avec $r > 0$ la distance de x à un point c définit des sphères centrées sur c de rayons différents. Avec une telle définition on peut décrire un nuage comme la surface décrite par une somme de fonctions de potentiel, pour un potentiel de référence donnée. Pour rendre les surfaces implicites plus simples d'utilisation, un squelette est souvent utilisé. Une isosurface est alors décrite par un triplé : un potentiel de référence, une fonction de potentiel $f(x)$ et une structure de contrôle qui peut être un point, un segment, un triangle...On obtient alors respectivement une sphère, une gélule ou un prisme arrondi aux bords. Ces primitives peuvent être combinées de manière à produire la forme souhaitée.

1.2 Échantillonnage par particules

Une fois la surface du nuage définie, convenablement, par un squelette, il faut pouvoir l'afficher. Cependant, l'expression analytique d'une isosurface est complexe, voire impossible à déterminer. De plus, le traitement par informatique ne s'accommode que très mal d'une expression purement analytique, il faut donc échantillonner la surface et l'approximer par un maillage triangulaire. Pour autant, cet échantillonage ne doit pas être refait à chaque déformation de la surface, car cette solution serait trop coûteuse, et non interactive. On peut trouver une solution à ce problème en utilisant un système de particules, qui se répand sur la surface, comme décrit dans l'article de Andy Witkin et Paul Heckbert [WH94]. La méthode proposée dans cet article est la suivante. On dispose sur la surface à échantillonner des particules dont l'énergie est définie par l'expression :

$$E^{ij} = \alpha \exp\Big(-\frac{|r^{ij}|^2}{2\sigma^2}\Big)$$

avec $r^{ij} = p^i - p^j$ le vecteur entre les particules i et j, α la constante globale qui définit l'intensité de répulsion entre les particules, et σ le rayon au delà duquel les particules ne se repoussent plus. L'expression de l'énergie d'une particule s'exprime donc comme la somme des interactions avec ses voisines :

$$E^i = \sum_{j=1}^{n} E^{ij}$$

Ainsi, chaque particule tend vers son minimum local d'énergie en migrant sur la surface pour s'éloigner des autres particules, son vecteur vitesse s'oriente donc dans la direction faisant décroître son énergie le plus rapidement. Une fois l'équilibre atteint[2], chaque particule se divise en deux, le paramètre σ étant divisé par $\sqrt{2}$, cette division se produit jusqu'à un certain niveau défini à l'avance. Cette méthode de répulsion simple est présentée dans l'article sous le titre de *Simple répulsion* cette méthode n'est utilisable que pour des surfaces à déformation lente et ne permet pas de prendre en compte des changements de forme important. Une autre méthode plus efficace est détaillée par la suite et utilise la répulsion adaptative, en spécifiant pour chaque particule un rayon de répulsion σ^i. Cette méthode permet de prendre en compte des déformations importantes et réagit plus rapidement aux changements de topologie, de l'isosurface, cependant pour le cas qui nous concerne elle n'apporte pas d'avantage d'information.

Un autre point important exposé dans cet article est l'expression de la contrainte qui oblige la particule à rester sur la surface implicite pour un potentiel donné. Son expression est particulière car elle n'agit pas sur la position de la particule mais sur sa vitesse. Il s'agit de projeter la vitesse courante de la particule sur l'isosurface, en utilisant le gradient calculé pour la position courante de la particule. l'expression de la vitesse de la particule i sous la contrainte est donc :

$$\dot{\mathbf{p}}^i = \mathbf{P}^i - \frac{F^i_{\mathbf{x}} \cdot \mathbf{P}^i + F_{\mathbf{q}} \cdot \dot{\mathbf{q}} + \phi F^i}{F^i_{\mathbf{x}} \cdot F^i_{\mathbf{x}}} F^i_{\mathbf{x}}$$

Cette expression est reprise telle quelle de l'article de Witkin et Heckbert, $F^i_{\mathbf{x}}$ représente le gradient associé à la particule i au point x, P^i la vitesse courante de la particule, F^i est la valeur de la fonction de potentiel au point x[3], la constante ϕ sert pour limiter les

[2] La dérivée de l'énergie inférieure à un certain seuil.

[3] Le potentiel est calculé par rapport à la structure de contrôle comme défini au paragraphe 1.1

erreurs d'intégration numérique engendrées par l'absence d'état initial stable. Les derniers termes $F_{\mathbf{q}}$ et $\dot{\mathbf{q}}$ sont respectivement : la dérivée de la fonction de potentiel et la dérivée des paramètres du squelette. Ces deux derniers coefficients permettent de prendre en compte dans la contrainte, les déformations de la surface.

1.3 Marching Cubes

Cet algorithme permet de représenter un volume de données, défini par un champ de potentiel. Son principe de base est dans l'ensemble assez simple. L'idée présentée par Lorensen dans [LC87], consiste à subdiviser le volume que l'on souhaite représenter en petits cubes qui forme une grille tridimensionnelle, on fixe alors une valeur de potentiel référence, par laquelle va passer la surface. En déterminant les intersections entre les cubes et la surface, on obtient une approximation sous la forme d'un maillage[4] ce qui permet de l'afficher. La difficulté de cet algorithme provient de la nécessité de traiter de manière indépendante chaque cas. Il existe en effet 256 façons de couper un cube avec une surface courbe. En partant de cette remarque, il suffit de placer les 256 combinaisons dans une table, et de parcourir la grille en se reportant à la table pour déterminer comment relier les intersections entre la surface et les cubes. Cet algorithme est souvent utilisé pour l'affichage de donnée issue d'IRM, mais il est possible de l'utiliser aussi pour afficher des surfaces implicites, puisqu'elles sont définies avec des fonctions de potentiel sur $\mathbb{R}$.

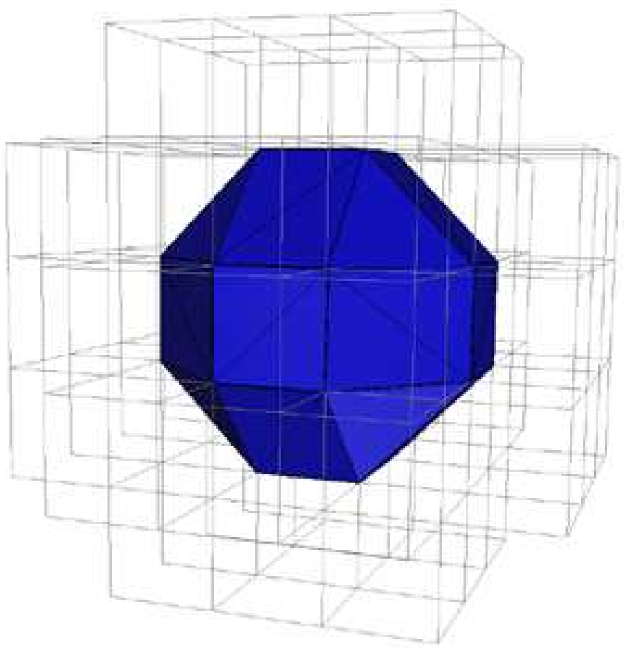


FIG. 3 – Exemple d'application de Marching Cubes, avec grille et surface

[4]Maillage constitué de triangle.

2 Présentation de la méthode utilisée

Je vais présenter la méthode que j'ai utilisée pour réaliser la triangulation dynamique, d'une surface implicite. Je me suis basé sur l'article de Hans-christian Rodrian et Hardy Moock [cRM96], qui reprend les idées de l'article de Witkin et Heckbert, dans sa majeure partie, puisque la représentation des surfaces implicites est faite par squelette ainsi que certains raisonnements. Cependant, cette méthode donne un résultat plus satisfaisant puisque l'on obtient un maillage ce qui n'est pas le cas de la méthode par particules. Je détaillerai la méthode présentée dans cet article, comment je l'ai implémentée, et pour finir les résultats que j'obtiens.

2.1 Détail de l'article

Le système masses-ressorts

Cette méthode est très physique dans le sens qu'elle utilise un système masses-ressorts, en effet, chaque point du maillage qui approxime la surface implicite est associé à une masse et chaque arête reliant deux points est un ressort. Cette disposition diffère de la méthode par particules puisque, l'on ne considère plus seulement une force du répulsion entre les points, mais également une force d'attraction. En effet, les ressorts suivent la loi de Hooke qui prend en compte la longueur au repos, la raideur et l'amortissement d'un ressort. La force appliqué sur les extrémités d'un ressort peut donc être calculée par la formule suivante :

$$f_r = -\left[k(||x_a - x_b|| - r) + \xi(v_a - v_b)\frac{x_a - x_b}{||x_a - x_b||}\right]\frac{x_a - x_b}{||x_a - x_b||}$$

avec x_a et x_b les coordonnées des extrémités a et b du ressort, r représentent la longueur au repos du ressort, k et ξ sont respectivement la constante de raideur et la constante d'amortissement. Les constantes k et ξ sont communes à tous les ressorts et doivent être choisie avec soin car le comportement du système masses-ressorts en dépend. Le choix de ressorts faiblement amortis (ξ faible) favorise une oscillation du maillage et entraîne une superposition de certains triangles, de même des ressorts trop mous ralentissent le système, l'équilibre étant atteint plus lentement. La force f_r est donnée négative ici mais on a évidemment $f_{ra} = -f_{rb}$. Suivant le même raisonnement que dans l'article de Witkin et Heckbert, chaque masse est soumise à la résultante des forces qui s'y appliquent. Donc pour une masse i donnée on obtient :

$$f_{ri} = \sum_{j=0}^{n} f_{rj}$$

Ainsi les masses se déplacent selon la direction qui fait décroître le plus rapidement cette somme pour minimiser l'énergie associée, tout comme avec les particules. Cette tendance naturelle des ressorts à tendre vers leur longueur de repos, entraîne des conséquences intéressantes aux niveaux de la triangulation. Si l'équilibre des forces est atteint, chaque masse est à égale distance de chacune de ses voisines. Il en résulte l'obtention d'un maillage très régulier. L'équilibre énergétique des ressorts ne peut être atteint que si le nombre de ressorts évolue dans le temps en fonction : de la taille de la surface et de l'état du système masses-ressorts *i.e* au repos ou pas. Cette dernière remarque, amène la nécessité de pouvoir supprimer ou ajouter des triangles, à volonté.

Les étapes de la méthode

D'une manière schématique il est possible de décomposer cette méthode en deux étapes principales :

1. *Création du maillage initial :*
   (a) *Calcul d'une triangulation initiale de la surface implicite.*
   (b) *Création du système masses-ressorts, (point$\Leftrightarrow$masse et arête$\Leftrightarrow$ressort).*
   (c) *Lancement de la mise à jour de la géométrie, par action du système masses-ressorts.*

   Une fois l'équilibre atteint il est possible d'animer la surface en déplaçant ou en modifiant les paramètres du squelette.
2. *Animation de la surface*
   (a) *Mise à jour de la géométrie par action du système masses-ressorts.*
   (b) *Animation de la surface.*

La mise à jour de la géométrie est l'étape la plus longue, puisqu'elle consiste d'abord à résoudre les équations différentielles et à appliquer la contrainte sur la vitesse des masses. La résolution des équations différentielles pose un petit problème, normalement il est nécessaire de résoudre des équations du second degré, car il faut déterminer la position, la vitesse et l'accélération, il est cependant possible de réécrire les équations différentielles sous la forme d'un système du premier degré comme ci-après :

$$\begin{aligned} v &= \frac{dx}{dt} \\ a &= \frac{dv}{dt} \end{aligned}$$

Elles sont résolues numériquement, en utilisant un schéma d'intégration *midpoint*, ou *Runge-Kutta*. L'accélération s'obtient simplement en appliquant la loi de Newton sur la résultante des forces pour chaque masse i, $a_i = \Sigma f_j / m_i$. La valeur de la masse n'étant pas un paramètre important, posée $\forall i, m_i = 1$, est correcte. Après la résolution des équations différentielles, il faut appliquer la contrainte sur la vitesse, de la même manière que dans l'article de Witkin et Heckbert et ainsi contraindre les masses à rester sur la surface. D'un point de vue numérique cette étape est délicate, puisque l'écart entre la vitesse obtenue par la résolution des équations différentielles et celle qui résulte de l'application de la contrainte peut être importante. En effet, si l'on considère le schéma d'intégration de *Runge-Kutta* :

$$\begin{aligned} k_1 &= hf(x_n, y_n) \\ k_2 &= hf(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}) \\ k_3 &= hf(x_n + \frac{h}{2}, y_n + \frac{k_2}{2}) \\ k_4 &= hf(x_n + h, y_n + \frac{k_3}{2}) \\ y_{n+1} &= y_n + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} + O(h^5) \end{aligned}$$

L'utilisation d'un pas de temps h élevé engendre un écart important et donc une instabilité qui risque de faire littéralement exploser le maillage. Une solution pour limiter cette erreur est d'appliquer la contrainte deux fois, une fois en $h/2$ (point 2 sur le schéma 4) et l'autre en h (position 4 sur le schéma 4). En diminuant ainsi l'erreur on maintient la stabilité du système masses-ressorts tout en gardant un pas de calcul élevé, les masses restent ainsi très mobiles sur la surface ce qui contribue à atteindre l'équilibre plus rapidement.

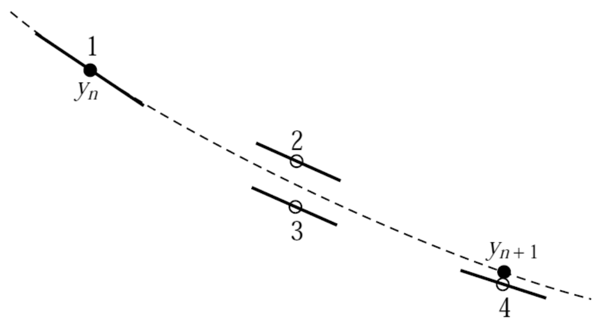


FIG. 4 – Position des points lors de l'évaluation
avec la méthode de Runge-Kutta d'ordre 4

L'étape suivante est l'application de règles simples pour régénérer les triangles du maillage, comme mentionné dans le paragraphe 5.2.2 de l'article [cRM96], deux cas posent problèmes :

- Si un ressort est trop compressé.
- Si un ressort est trop étiré.

Par rapport à l'article, cette étape est très discutable car on y suppose que la surface est une *2 variété*, c'est à dire que le cardinal de l'intersection du voisinage de deux points doit être égal à deux, ce qui n'est en rien garanti puisqu'il n'y a aucune hypothèse sur la triangulation initiale. Les autres cas sont ignorés pour le moment et seront détaillés plus tard.

La détection d'un ressort trop compressé et devant être supprimé est faite selon la méthode suivante :

$$\frac{|r^{ij}|}{R} < D_{min}$$
$$|r^{ij}| - |r^{ij}|_{old} < \varepsilon_{length} R, \qquad \varepsilon_{length} \geq 0,$$

Où rij est bien sûr, la distance entre les deux extrémités du ressort et D_{min} est une constante fixée qui définit la compression maximale d'un ressort. Si un ressort est trop compressé et ne tend pas à s'allonger, il sera supprimé et son voisinage sera mis à jour selon le schéma suivant :

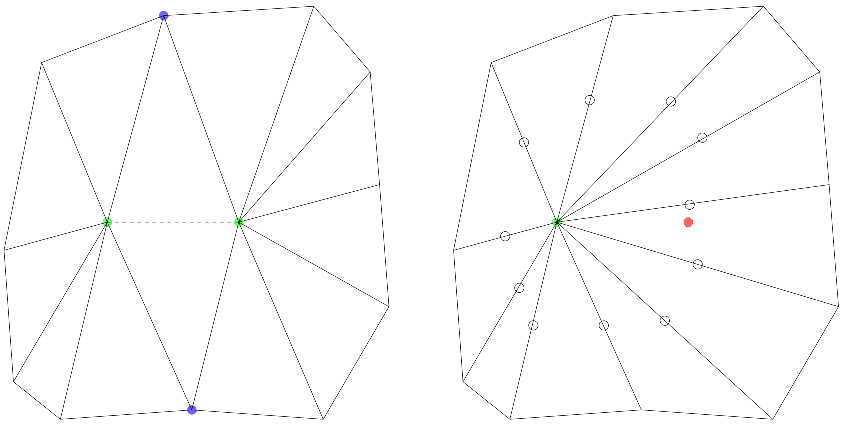


Fig. 5 – Détail de la mise à jour d'un ressort compressé

La détection d'un ressort trop étiré devant être divisé en deux est faite par la méthode suivante :

$$\frac{|r^{ij}|}{R} > D_{max}$$
$$|r^{ij}|_{old} - |r^{ij}| < \varepsilon_{length} R, \qquad \varepsilon_{length} \geq 0,$$

D_{max} est une constante qui définit l'étirement maximal d'un ressort. Si un ressort est trop étiré et ne tend pas à rétrécir, il sera divisé en deux, et son voisinage sera mis à jour selon le schéma suivant.
En plus de la suppression ou de l'ajout de ressort, tous les ressorts voisins de ceux mis à jour sont marqués[5] et ne seront pas pris en compte lors de la prochaine mise à jour de la géométrie, afin d'éviter les oscillations entre destruction et création des ressorts.

[5]Voir les petits ronds sur les schéma 17

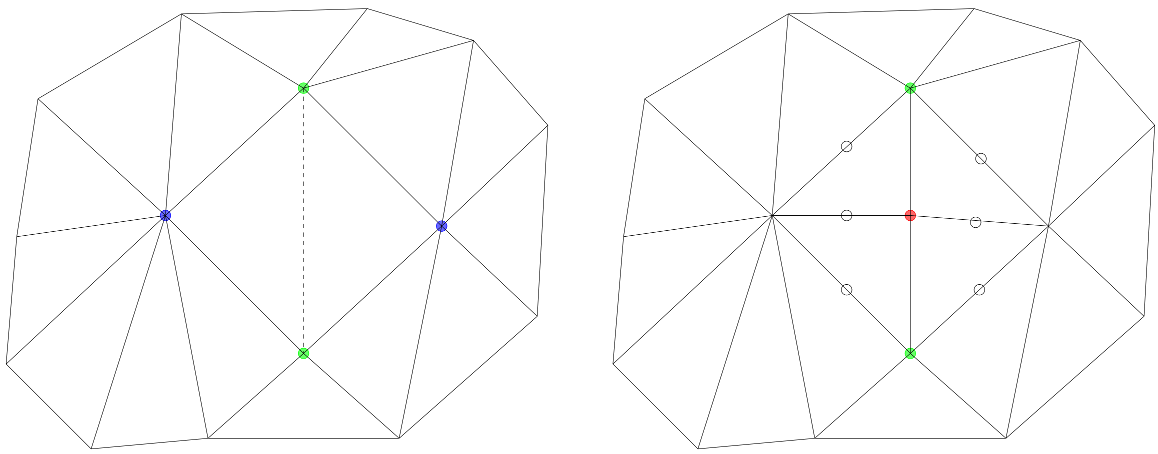


FIG. 6 – Détail de la mise à jour d'un ressort étiré

Traitement géométrique

Dans cette partie je vais présenter une partie qui ne figure pas dans l'article et qui n'a pas été implémentée avec succès. Ce paragraphe traite des méthodes qui pourraient être utilisées pour résoudre les problèmes lorsque des cas de triangulations dégénérés apparaissent. Pour plus de clarté j'ai choisi une notation proche de celle utilisée en graphe puisque comme détaillé 2.1 page 14, la mise à jour du maillage utilise beaucoup de voisinage. Ainsi, par la suite je prendrais les conventions suivantes :

- $\Gamma_s(i)$ représente le voisinage d'un point, i.e tous les points qui forment avec i un ressort.
- $\Gamma_r(i)$ représente le voisinage d'un ressort, i.e les ressorts adjacents aux extrémités de ressort.
- $\Gamma_t(i)$ représente tous les triangles qui ont i pour sommet.

Avec une telle notation la propriété de *2 variété* donne $|\Gamma_s(a) \cap \Gamma_s(b)| = 2$, i.e le cardinal de l'intersection de voisinage vaut deux. Si l'on considère un cas comme la figure 10 page 21 le cardinal vaut trois et donc il n'est plus possible de déterminer les points extérieurs qui appartiennent aux bords. L'idée est donc d'exploiter les particularités du point central. Considérons les figures suivantes :

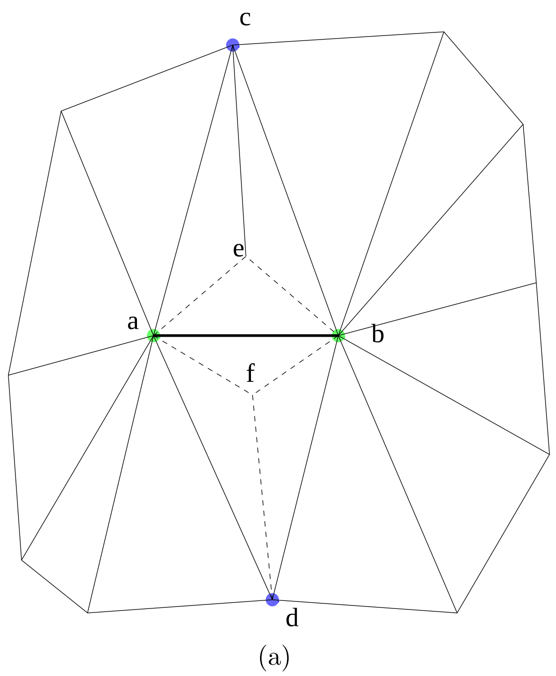


FIG. 7 – Illustrations de cas dégénérés

Dans le cas (a) il est possible d'écrire la relation suivante : $|\Gamma_s(a) \cap \Gamma_s(b)| = 4$, posons alors $S = \Gamma_s(a) \cap \Gamma_s(b) = \{c, d, e, f\}$, il est alors possible de déterminer dans S les points qui n'appartiennent pas aux bords i.e e et f. En effet, il est possible dans un cas comme celui-ci de les définir par : l'ensemble des $i \in S$ tq $\Gamma_s(i) \subset S \cup \{a, b\}$. Sachant qu'il n'y a pas d'intersection entre les triangles, seuls les points intérieurs appartiendront à cet ensemble. La détection de ces points permet de supprimer les triangles et les ressorts, qui contiennent ces points. Grâce à cette définition il est possible de récupérer la propriété de *2 variété* sur cette portion du maillage. Cette méthode marche bien si l'on reste sur des cas simples, pourtant elle pose des problèmes si les triangles sont subdivisés de manière plus profonde comme par exemple sur le schéma (b) de la figure 7 page 18 car on a toujours $|\Gamma_s(a) \cap \Gamma_s(b)| = 4$, mais la propriété d'inculsion de l'ensemble n'est plus vraie, car $\Gamma_s(e) \not\subset S \cup \{a, b\}$. La présence de ces cas particuliers limite l'utilisation de ce critère pour résoudre les dégénéréscences de la triangulation, ce qui explique qu'il n'ait pas été utilisé dans l'application finale. La solution serait sans doute d'utiliser les dimensions supérieures i.e de travailler sur les arètes ou les triangles avec Γ_t et Γ_r.

2.2 L'implémentation

La structure de données

La partie la plus difficile de l'implémentation est le choix de la structure de données puisqu'en dépend la complexité de la méthode. Certes la résolution numérique des équations différentielles restera en $C_1 \times O(n)$, n étant le nombre de ressorts. La constante C_1 dépend du choix de la méthode de résolution et peut aller de un pour Euler implicite, jusqu'à six pour England. Une autre complexité indépendante de la structure de données est l'évaluation de la fonction de potentiel et de son gradient. J'ai choisi pour les squelettes composés d'une fonction de potentiel semblable à celle utilisé pour les *metaballs* $a \exp^{-br^2}$, et pour ceux constitués d'une ligne une fonction similaire à la distribution de Cauchy $\frac{1}{(1+b^2r^2)^2}$. Dans les deux cas a et b sont des paramètres de contrôle qui détermine la décroissance des fonctions de potentiel et r la distance euclidienne d'un point au squelette voir l'article [JT02] pour le détail des fonctions de potentiel et voir 8 page 19 pour des exemples de fonctions de potentiel. Dans le détail de l'algorithme au paragraphe 2.1 page

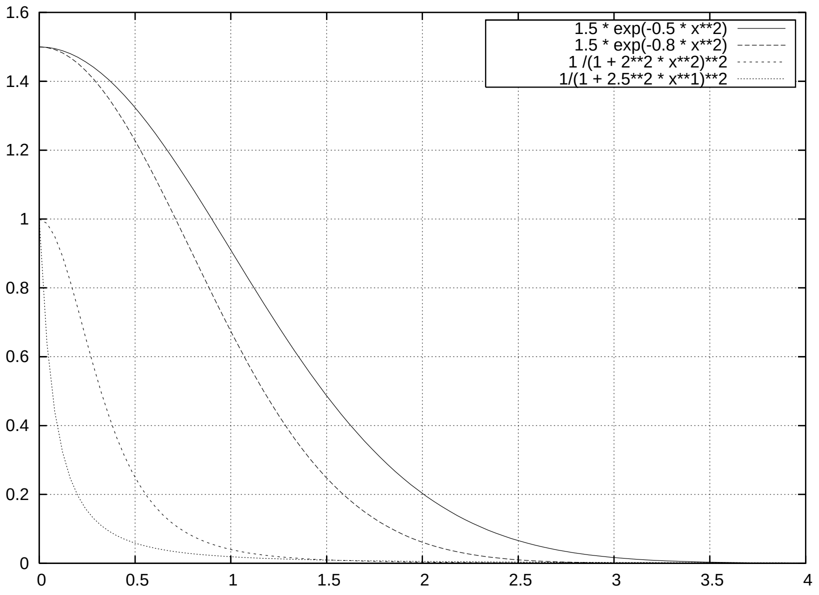


Fig. 8 – Différentes fonctions de potentiels

14, j'ai mentionné la nécessité de calculer les points adjacents à un ressort, ainsi toute la structure de données est optimisée pour la gestion des relations de voisinages entre 3 types d'éléments :

- les points
- les ressorts
- les triangles

À partir de trois points, il faut donc pouvoir, déterminer le triangle qu'ils forment, et à partir de deux points déterminer le ressort, et réciproquement. Les opérations ensemblistes d'intersection, d'union, et d'appartenance, peuvent être calculées linéairement si l'on représente les ensembles par des arbres équilibrés. On peut alors connaître quel est le triangle formé par trois points en calculant deux intersections. Comme on a pu le voir dans le paragraphe 2.1 page 13, le maillage évolue constamment, de nombreux ressorts et triangles sont ajoutés dynamiquement. Pour permettre un maximum d'efficacité, la structure de données est organisée à l'aide d'arbres et de piles. Trois arbres sont utilisés :

- pour les triangles, qui stockent les index des trois points.
- pour les ressorts qui stockent les index de deux points.

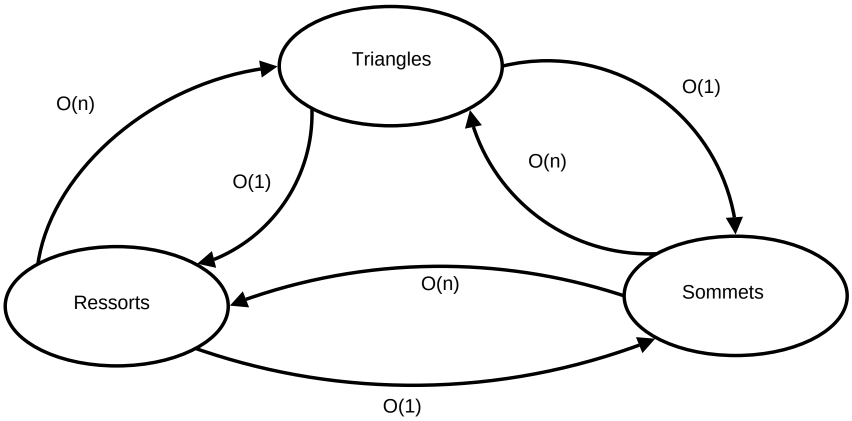


FIG. 9 – Complexité des passages d'un mode de représentation à l'autre

– pour les points qui stockent les relations d'adjacence, sur les triangles et les ressorts.

Pour chaque arbre une pile permet d'allouer un identifiant unique pour chaque triangle, ressort et point. Ainsi l'ajout d'un élément qu'il s'agisse d'un point, d'un triangle ou d'un ressort, s'effectue de la même manière :

1. Tester la pile pour savoir si un identifiant est libre, ou incrémenter le compteur associé au type de l'élément si la pile est vide.
2. Trouver les identifiants des sommets pour un triangle, ceux des extrémités pour un ressort (si l'on ajoute un sommet cette étape est sautée)
3. Puis insertion dans l'arbre qui s'équilibre.

Cette utilisation de simple entier pour identifier les éléments, limite l'utilisation de pointeurs, et permet de généraliser le principe, à tous les types d'éléments. Grâce a cette méthode la recherche d'un élément est en $\log(x)$ où x est le cardinal de l'ensemble concerné.

Complexité de la méthode

À partir de la description de la structure de données, une analyse de la complexité de la mise à jour de la géométrie du maillage est possible. Le plus simple est de travailler par rapport aux éléments principaux, ainsi pour chaque ressort que l'on met à jour il faut :

1. Calculer les deux points adjacents $O(m)$ avec m le nombre de voisins d'un point.
2. Si l'on doit supprimer un ressort
   – Calculer tous les ressorts adjacents à l'une des extrémités et les modifier $O(l)$ avec l le nombre de ressorts adjacent à un point
   – Mettre à jour les triangles ayant ce point pour sommet $O(p)$, p étant le nombre de triangles adjacents par sommets.
3. Si l'on sépare un ressort en deux
   – Ajouter les nouveaux ressorts et mettre à jour le voisinage des points $5 \times O(m)$.
   – Ajouter des triangles et mettre à jour les voisinages $4 \times O(p)$.

Dans la pratique on remarque que les triangles forment des motifs hexagonaux, ce qui donne une moyenne de six voisins pour un point, six ressorts et six triangles. Sachant que cette opération est réalisée pour chaque ressort, on obtient un complexité linéaire en fonction du nombre de ressorts à mettre à jour, la mise à jour des ressorts voisins et des triangles adjacents devenant alors une constante.
La complexité de cette méthode est donc de $O(n) + O(m)$, n étant le nombre de points et m le nombre de ressorts mis à jour à chaque itération.
Il est possible de diminuer le terme en $O(m)$ en utilisant des tables de hachage, mais si $n >> m$ ce qui est le cas en général, ce terme est négligeable.

Jusqu'à présent nous avions considéré les mêmes hypothèses sur les propriétés de la surface, que dans l'article, pourtant dans certaines configurations la surface n'est plus une 2 variété, en conséquence de quoi il n'y a plus deux points adjacents à un ressort. Par exemple dans le cas suivant :

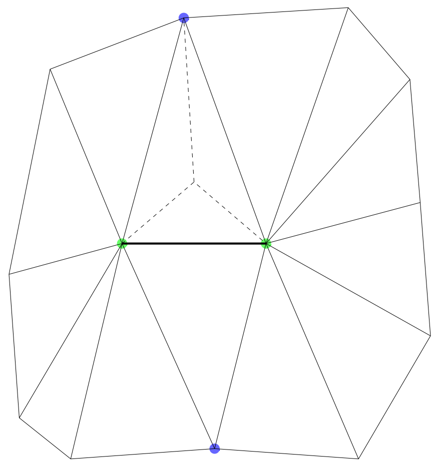


FIG. 10 – Exemple de surface localement dégénérée

Cet exemple de configuration pose de nombreux problèmes pour la méthode exposée ci-dessus, car le cardinal de l'intersection des voisinages de deux points n'est plus égal à deux. Dans ce cas, la mise à jour du maillage par la méthode précédente crée des trous dans la surface. Cette situation n'est pas mentionnée dans l'article, pourtant, les expérimentations ont montré qu'elle est très fréquente. En revanche on peut remarquer que ce cas sera résolu tôt ou tard puisque les ressorts internes (en pointillés) sont plus compressés que le ressort courant (en gras). La solution la plus simple est donc de sauter ce cas en laissant le réseau de ressorts absorber ce problème aux cours de itérations suivantes. Les expérimentations ont montré que même naïve cette méthode est efficace, les seules limitations qu'elle entraîne sont une perte notable de vitesse et un risque de blocage si de trop nombreux cas dégénérés se présentent en même temps. La bonne solution serait de choisir quelque cas que l'on traite en réparant la triangulation par suppresssion des triangles qui posent problème. Cette opération est assez délicate comme mentionné au paragraphe 2.1 page 17. La résolution de ces problèmes, liée à la géométrie permettrait de faire de nombreuses améliorations en faveurs des performances de l'application.

3 Résultats

3.1 Vitesse et performances

Dans le paragraphe 7 (*discussion*) de l'article les performances sont relativement faibles par rapport à celles que j'obtiens[6]. En effet, l'animation d'un squelette composé de 2 traits disposés en croix est réalisée en temps réel à plus de 30 images par sec, pour un maillage de 1200 triangles en moyenne. J'ai programmé cette méthode en C++ en utilisant le plus possible de conteneur STL se qui factorise énormément le code, de plus la mémoire est gérée au plus juste grâce aux smart pointeur de la bibliothèque Boost. L'utilisation de ces composants rend l'ensemble très stable et facilement modifiable. L'affichage utilise OpenGL, et glut comme gestionnaire de fenêtre.

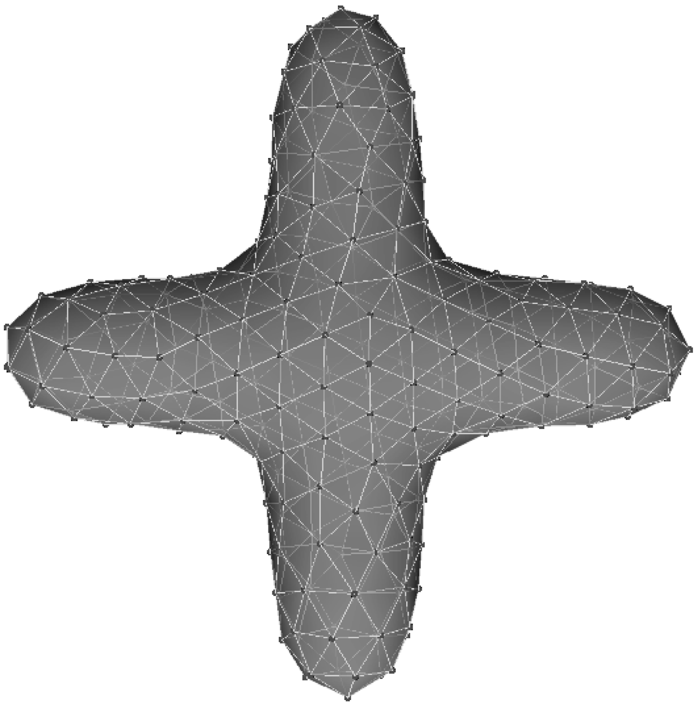


FIG. 11 – Exemple de maillage obtenu après mise à jour de la géométrie

Les vitesses obtenues permettent d'envisager l'animation de surfaces composées d'un plus grand nombre de triangles, cependant la vitesse de déformation de la surface devra être moindre, ce qui pour un nuage est très convenable. Parmi les résultats remarquables, la représentation par squelette permet d'ajouter à volonté d'autres éléments, il faut cependant que cela n'induise pas de changement de topologie. De plus, il faut garder le

[6] Il faudrait prendre en compte la différence de matériel pour être parfaitement exact.

nombre de composantes connexes constant. En effet, l'ajout d'un point de squelette extérieur à toute surface préalablement existante n'ajouterait rien. Il faudrait refaire une triangulation par Marching Cubes, pour générer la triangulation initiale, nécessaire à cette méthode.

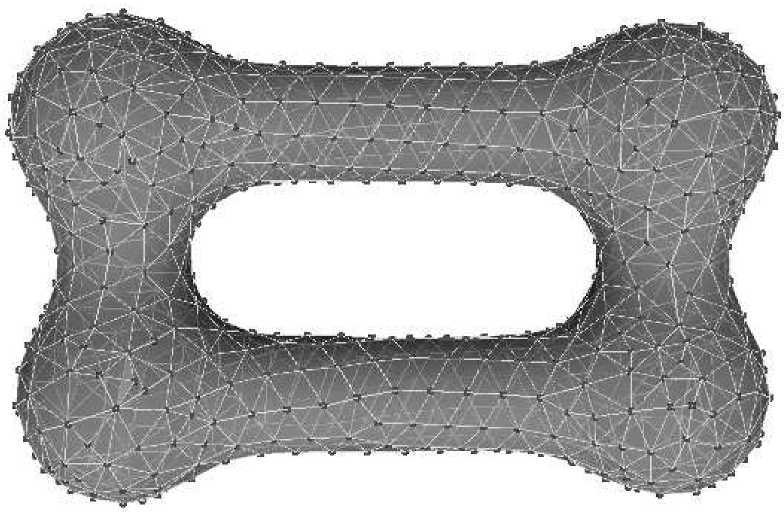


FIG. 12 – Exemple de maillage animé dans les coins

3.2 Limitations

Les principales limitations proviennent de la trop grande simplification du traitement géométrique, j'ai fait le choix de sauter les cas des triangles dégénérés, la surface n'étant alors plus une 2 variété. Cette simplification pose d'autres problèmes, que ceux mentionnés aux paragraphes 2.2, mais provoque aussi le retournement de certains triangles, causant localement la superposition très gênante de deux triangles. Ce problème apparaît de manière flagrante si la surface est laissée au repos.

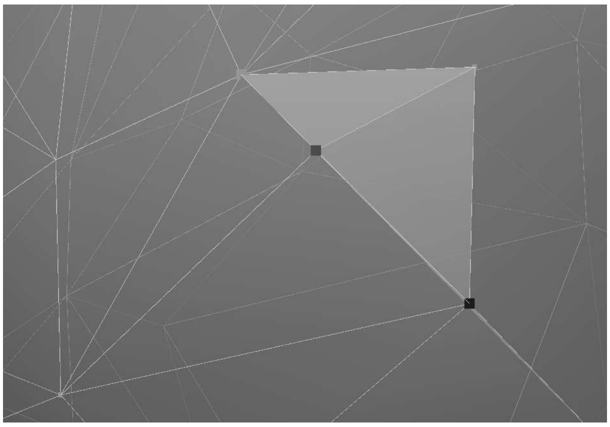


Fig. 13 – Exemple de replis d'un triangle

Confronté à ce problème pendant trois semaines j'ai cherché un certain nombres de critères permettant de réduire, et de récupérer les cas de triangles dégénérés, pourtant de trop nombreux cas sont à envisager. Une solution plus simple serait de réduire le nombre des cas que l'on souhaite traiter par exemple :

– les ressorts ayant trois points adjacents.
– les ressorts ayant quatre points adjacents 2 au dessus et 2 en dessous.

Parmi les autres limitations intervient la vitesse qui est liée à la taille du pas de calcul pour la résolution numérique des équations différentielles. En effet, la complexité du solveur est linéaire, mais le pas de calcul étant assez faible, les points n'atteignent leurs position final qu'après plusieurs itérations. Pour une raison de stabilité j'ai fait le choix d'utiliser un schéma de type Runge-Kutta, cependant l'utilisation d'un solveur plus robuste comme Verlet permettrait d'augmenter le pas de calcul, augmentant la vitesse des points et donc la déformation de la surface. Dans la partie 2.1 page 13 j'ai donné l'expression de la contrainte, qui permet aux points de rester sur la surface, cependant seule la partie liée au gradient à été implémenté, ainsi les déplacements du squelette ne sont pas prises en compte. Ce problème est un facteur limitant, par conséquent il n'est pas possible d'animer la surface par des mouvements violents et très rapides.

3.3 Autres solutions envisageables

Jusqu'à présent j'ai présenté la méthode décrite dans l'article de Hans-christian Rodrian et Hardy Moock, pourtant d'autre méthodes sont envisageables pour réaliser une triangulation dynamique de surfaces implicites. Par exemple, en utilisant une méthode par re-triangulation, basée sur un système de particules, l'idée principale est de re-trianguler seulement les zones qui en ont besoin. On utilise ainsi la cohérence temporelle, car lors de leur déplacement les particules bougent peu par rapport à leur position antérieure. Après avoir déterminé les zones de la surface qui doivent être re-trianguler, il est possible d'appliquer un algorithme de triangulation rapide, qui exploite cette propriété comme *Spiraling Edge* de Patricia Crossno [CA99] ou encore une méthode par projection comme celle de Gopi [GK00]. Un des problèmes de cette méthode est le coût de calcul du voisinage des particules qui doit être mis à jour lors de la déformation de la surface. De plus les deux algorithmes de triangulation ci dessus risquent de générer une triangulation moins régulière, [7] et donc moins adaptée pour le calcul d'un rendu de qualité. Pourtant cette solution n'est pas à écarter car le système de particules qui reste à la base de cette méthode est très robuste et supporte les déformations violentes, de plus les changements de topologie sont possibles, le passage d'une sphère à un tore ne pose à priori aucun problème.

Cette autre méthode pourrait être utile pour régler les problèmes de géométrie mentionner page 17. En effet, au lieu de déterminé comment résoudre le problème de triangulations, refaire la triangulation dans les zones dégénérées permettrait de résoudre les problèmes liés à la géométrie et donc de palier cette limitation.

Conclusion

Ce stage fut pour moi l'occasion de prendre contact avec des problèmes très variés et enrichissants, tout en abordant des domaines qui ne m'étaient pas familiés à priori. Les résultats obtenus sont assez satisfaisants, de plus cette méthode par nature permet d'obtenir un résultat qui est visuellement plaisant. D'autre part, il est possible d'envisager mettre en œuvre cette méthode dans d'autres domaines, puisque tout type d'application utilisant un potentiel scalaire peut utiliser cette méthode, par exemple avec des *level set* animés. Cependant certains problèmes restent ouverts, et peuvent être résolus dans un avenir proche, notamment les erreurs géométriques qui empèchent de réaliser des changements de topologie. De plus l'intégration des dérivées de la fonction de potentiel et du squelette dans la contrainte rendraient cette méthode encore plus rapide. D'un point de vue personnel, je tiens à remercier toute l'équipe ÉVASION pour son accueil, son ouverture d'esprit et sa gentillesse.

[7] Ils utilisent des critères locaux.

Références

[Bou04] Antoine Bouthors. Rendu réaliste de nuages en temps réel. Master's thesis, Université de Marne la Vallée, septembre 2004.

[CA99] Patricia J. Crossno and Edward S. Angel. Spiraling edge : Fast surface reconstruction from partially organized sample points. In *IEEE Visualization '99*, pages 317–324, San Francisco, 1999. IEEE.

[cRM96] Hans christian Rodrian and Hardy Moock. Dynamic triangulation of animated skeleton-based implicit surfaces, September 04 1996.

[GK00] M. Gopi and S. Krishnan. A fast and efficient projection-based approach for surface reconstruction, April 27 2000.

[JT02] Xiaogang Jin and Chiew-Lan Tai. Analytical methods for polynomial weighted convolution surfaces with various kernels. *Computers & Graphics*, 26(3) :437–447, 2002.

[LC87] W. E. Lorensen and H. E. Cline. A high resolution 3D surface construction algorithm. In *SIGGRAPH '87 Conference Proceedings*, pages 163–170, July 1987.

[WH94] Andrew P. Witkin and Paul S. Heckbert. Using particles to sample and control implicit surfaces. *Computer Graphics*, 28(Annual Conference Series) :269–277, July 1994.