Compte Rendu - TP - Animation de tissus

Di Folco Maxime - Girot Charly

15/12/2017

1 Introduction

L'objectif de ce te TP est de simuler les interactions d'un tissu avec son environnement extérieur. Cette simulation est réalisée par la modélisation de différentes forces de ressort et leur intégration temporelle par la méthode d'Euler explicite.

2 Présentation du modèle

Afin de simuler un tissu, chaque sommet du maillage est relié à ses voisins par des ressorts. Cette technique permet de simuler et de modéliser le comportement d'objets déformables.

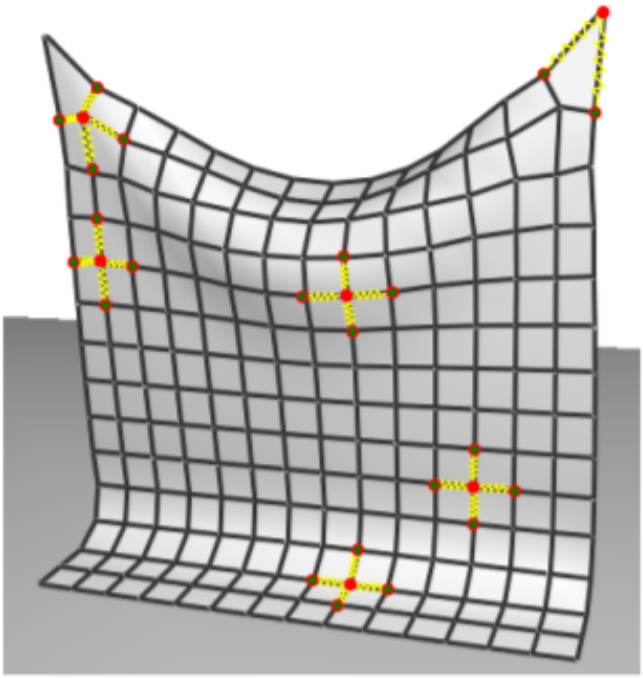

FIGURE 1 – Exemple de surface déformable formée par un ensemble de ressort

2.1 Étude d'un point relié à une masse

Code - tp_tissus_partie_1
Étudions tout d'abord le cas d'une masse fixe reliée à une autre masse mobile, par un ressort. Ce cas sera ensuite généralisé à chaque sommet du maillage.

En 3D, la force de rappel d'un ressort reliant deux points aux positions p_0 et p_1 est définie par la relation suivante :

$$F(t) = K(L_0 - \|(p_1 - p_0)\|)\frac{p_1 - p_0}{\|(p_1 - p_0)\|} \tag{1}$$

avec K le coefficient de raideur du ressort et L_0 la longueur à vide du ressort.

Cette masse subit également la force de gravité et la force de frottement fluide définies par :

$$F_g = -mg, F_d = -\mu v \tag{2}$$

En utilisant l'équation de mouvement dans sa version discrétisée, suivant la méthode dite d'Euler explicite et en considérant que la masse de l'objet est égale à 1, on obtient l'équation suivante :

$$\begin{cases} v(t + \Delta t) = (1 - \mu\Delta t)v(t) + \Delta t(F(t) + g) \\ p(t + \Delta t) = p(t) + \Delta t \end{cases} \tag{3}$$

C'est grâce à cette équation et en choisissant un Δt approprié (cf paragraphe suivant) qu'une évolution du système pourra être observée au cours du temps.

(a) (b)

FIGURE 2 – Système étudié en mouvement (a) et à l'équilibre (b)

L'inconvénient de la méthode d'Euler explicite est qu'elle diverge très facilement. Le choix de la valeur de Δt est donc très important, pas trop grand pour éviter que le système diverge, ni trop petit pour que le rendu soit le plus réaliste possible c.a.d pas trop lent comme si les forces ne s'exerçaient pas. De plus, un coefficient de raideur K trop grand implique que le ressort se comporte selon une relation quasi-rigide et le système diverge. A l'inverse un K trop faible, entraine un rendu non réaliste et un ressort se comportant tel un élastique. Quant au coefficient μ, une valeur trop petite provoquera l'effet que le tissu tombe sans frottements et une valeur trop grande l'impression que le tissu tombe anormalement lentement. Une valeur trop faible provoque également la divergence du système.

Après avoir testé plusieurs jeu de paramètres, pour la suite du TP les valeurs des paramètres seront fixées. Elles pourront être légèrement modifié si la qualité du rendu le nécessite ou tendent à faire diverger le système.

Ensuite, une autre masse est ajoutée, reliée à la seconde par un second ressort. Une chaine de deux ressorts couplés est ainsi créée.

(a) (b)

FIGURE 3 – Système étudié avec 2 masses en mouvement (a) et à l'équilibre (b)

Lors de la modélisation de la chute d'un tissu, chaque sommet sera couplé de la même manière à plusieurs ressorts.

2.2 Généralisation à un tissu "maillé"

Code - tp_tissus_partie_2
Pour chaque sommet du maillage, trois types de forces de ressort sont calculées :

- Les forces structurelles, liants les 4 voisins directs d'un sommet. Ce sont ces ressorts qui donnent la structure principale du tissu.
- Les forces de cisaillement (shear), liants les 4 voisins diagonaux les plus proches du sommet. Ces ressorts limitent les déplacements diagonaux des faces du maillage.
- Les forces de courbure (bend) liants les voisins espacés de 2 sommets. Ces ressorts limitent la courbure de la surface lors de sa déformation.

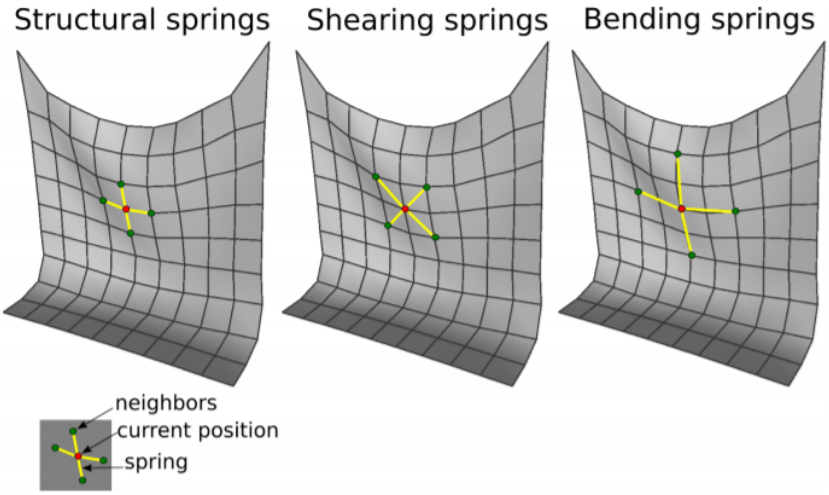

FIGURE 4 – Type de ressort appliqués au point courant (cercle rouge)

Pour gérer au mieux l'implémentation de ces trois forces et gérer les différents cas de frontières selon la force appliquée, nous avons avons utilisé l'algorithme suivant :

Algorithm 1 Algorithme de calcul des forces

```
for all points du maillage do
  Initialisation d'un vector qui a pour but de sauvegarder des forces
  for all forces de ressorts do
    for all voisins de notre sommet do
      if Le voisin appartient au maillage then
        return Calcul de la force et sauvegarde dans l'array des forces
      else
        Sauvegarde dans l'array des forces du vecteur (-1,-1,-1)
      end if
    end for
  end for
  for all forces dans l'array do
    if la force est différente de (-1,-1,-1) then
      return On rajoute la force au sommet
    end if
  end for
  On vide notre array
end for
```

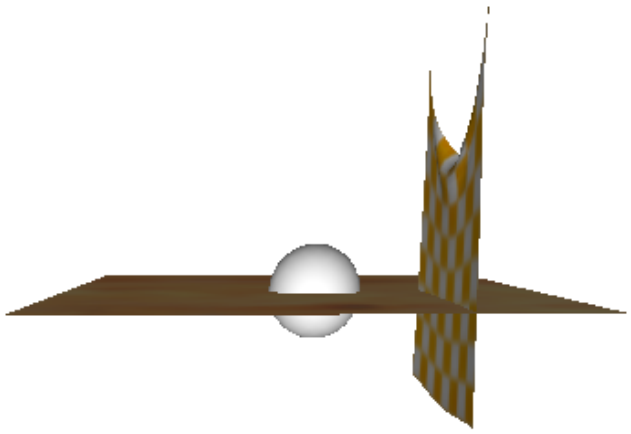

FIGURE 5 – Implémentation des forces des ressorts de notre maillage

La stabilité du système dépend du choix des paramètres. Les valeurs précédemment choisies sont affinées afin que le système soit stable. La collision entre le tissu et son environnement (maillage sol et maillage sphère) n'est pas gérée. Cette implémentation fera l'optique de la section suivante.

3 Gestion des collisions

Pour faire interagir la tissu avec son environnement, deux types de collisions sont implémentées :
- collision avec un plan
- collision avec une sphère.

3.1 Collision avec un plan

Pour gérer la collision avec un plan, la position de chaque point du maillage est comparée à la hauteur du plan. Si le point se trouve en dessous, la position de ce point est alors re-située au point d'intersection, majorée d'une erreur très faible, pour un rendu réaliste sans divergence. Les composantes orthogonales à l'intersection de la force et la vitesse sont alors annulées.

Dans notre exemple, notre plan est horizontal. Lorsqu'il y a collision nous mettons donc à 0 la composante z des forces et de la vitesse.

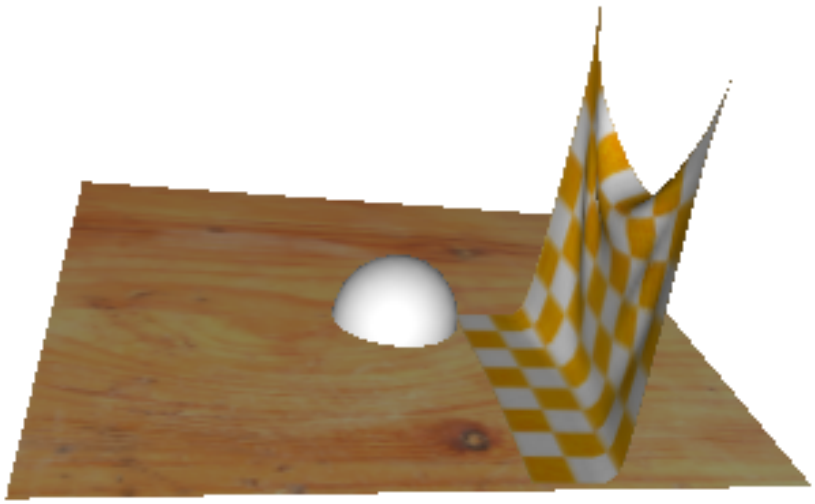

FIGURE 6 – Collision entre notre tissu et notre plan

3.2 Collision avec une sphère

Pour tester la collision entre une sphère et le maillage, il est vérifié que la distance entre chaque point du maillage au centre de la sphère, est supérieure au rayon de la sphère.

Dans le cas de la sphère, les collisions ne sont plus orthogonales. Il faut alors calculer la normale et soustraire les forces et vitesses selon l'axe de la normale au point de collision. Cela se traduit mathématiquement par la relation suivante :

$$\begin{cases} v^{k+1} = v^k - \langle v^k, n \rangle n \\ f^{k+1} = f^k - \langle f^k, n \rangle n \end{cases} \tag{4}$$

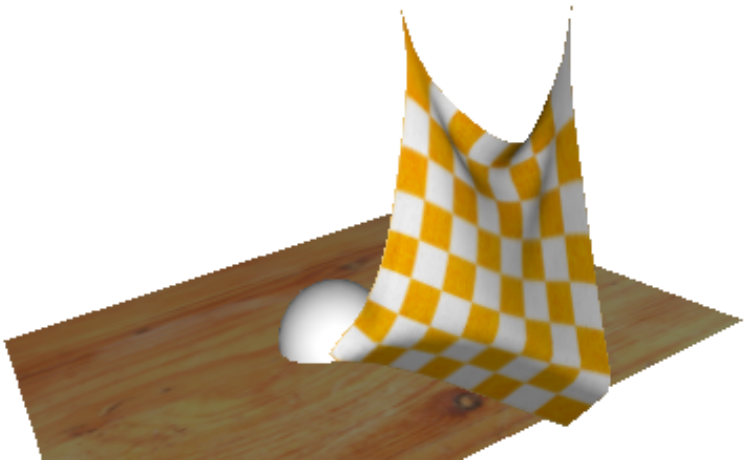

FIGURE 7 – Collision avec une sphère

3.2.1 Création du vent

Le vent a ensuite été modélisé pour donner du mouvement au tissu, grâce au modèle suivant :

$$F_w = K_w \langle a, n \rangle n \tag{5}$$

avec K_w une constante correspondant à l'intensité du vent, n la normale à la surface et a la direction du vent.

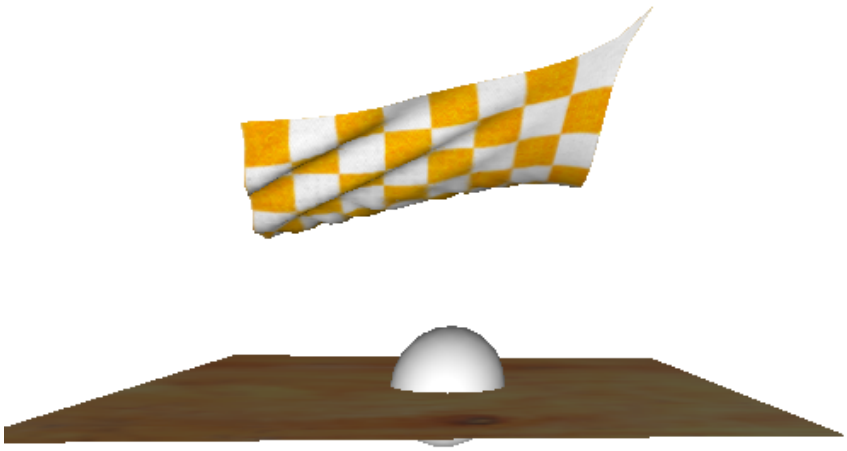

FIGURE 8 – Photo de la cape de l'homme invisible entrain de voler

4 GPU

Code - tp_tissu_gpu Cette modélisation est également effectuée sur GPU. Les positions, vitesses et normales des sommets sont passées du CPU au GPU comme des textures. Ces textures sont modifiées au sein de fragment shader qui gèrent également les collisions.

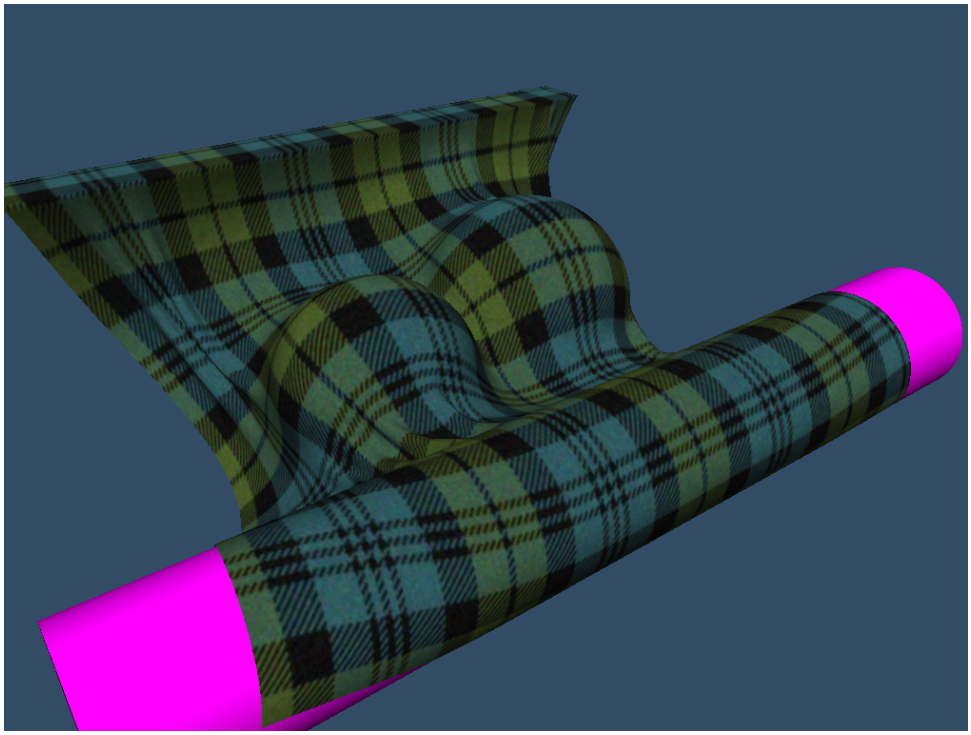

FIGURE 9 – Scène implémentée sous GPU

Nous avons mesurer le temps que prenait 100 itérations de la fonction d'affichage (draw_scene pour CPU et display_callback pour GPU). Pour le CPU nous obtenons un temps $t_{CPU} = 1300$ ms et pour le GPU un temps $t_{GPU} = 60$ ms. Comme attendu la compilation sur GPU est beaucoup plus rapide. Il est donc possible de faire un nombre plus important de calculs de forces ou de collisions.

En ce qui concerne l'implémentation des deux méthodes en général, nous pensons que le calcul des déformations est plus facile à implémenter dans les shaders de la méthode de GPU car ils sont traités point par point, alors que sur CPU, la gestion des frontières a été plus difficile à mettre en place. Néanmoins, nous avons préféré utilisé le CPU car la mise en place du code reste plus accessible et plus facilement compréhensible que sur GPU où la syntaxe devient parfois lourde.

5 Amélioration

Code - tp_tissus_perso

Pour que l'utilisateur puisse étudier l'influence des paramètres sur notre modèle, nous avons rajouter la possibilité de modifier interactivement les paramètres Δt notre pas de temps, F_w l'intensité du vent, et les trois coefficients de raideurs de nos ressorts (structural, shearing, bending). Nous avons aussi rajouter la possibilité de modifier le nombre de points fixes de notre tissu.

Certains choix de valeur font diverger le système.

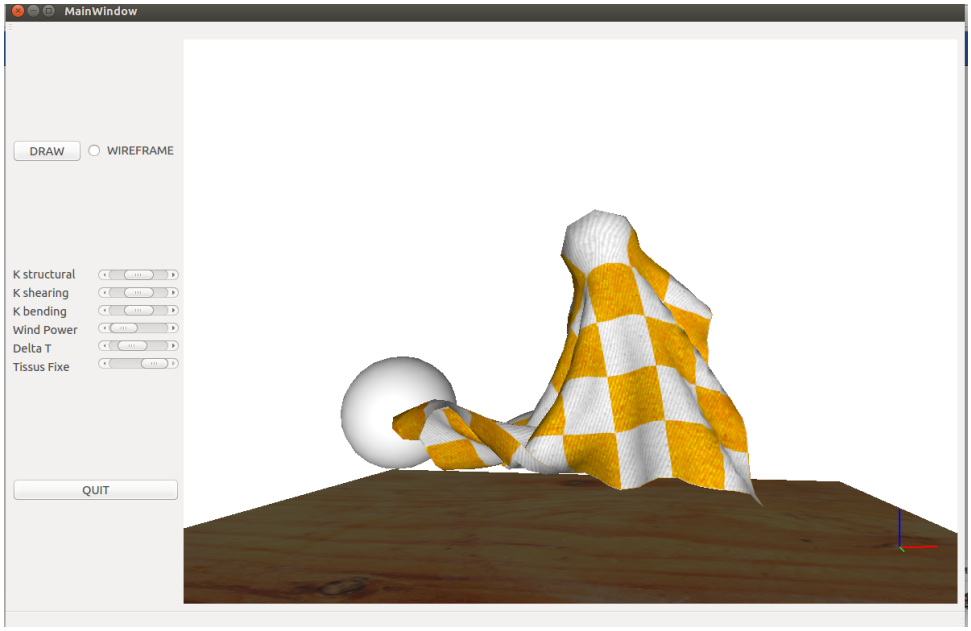

FIGURE 10 – Fenêtre utilisateur et exemple de collision avec le chat

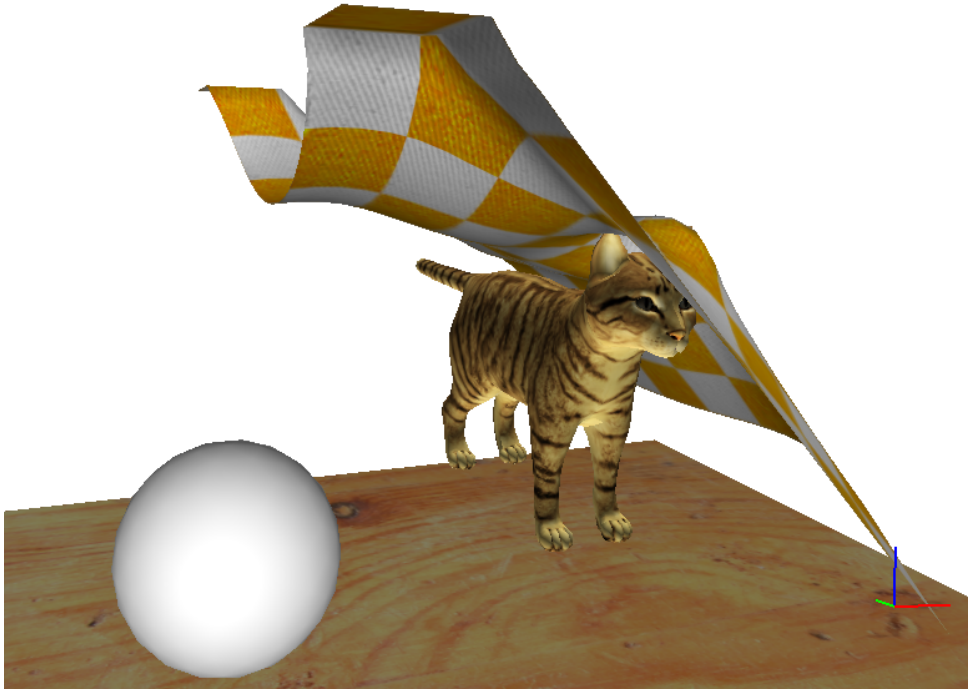

FIGURE 11 – Exemple de collision avec le chat et d'attachement du tissu

Pour rendre notre scène plus intéressante, nous rajoutons un chat à notre scène. Nous essayons d'ajouter notre tissu sur le dos du chat pour gérer un cas de collisions plus complexe qu'un plan et une sphère.

En réalité, pour simuler la collision, le chat est representé par un ensemble de sphères et de cylindres avec les quels la collision est simple à gérer. Les collisions avec les cylindres sont calculés d'une manière

similaire aux collisions avec une sphère à l'exception que le centre d'intérêt n'est pas fixe mais le long de l'axe du cylindre.

Dans le cas idéal, il faudrait calculer l'enveloppe convexe (Quick Hull 3D TP2) et utiliser cette enveloppe convexe pour gérer les collisions. L'implémentation d'une telle méthode n'a pas été réalisée.