# Rapport de la partie : “Animation de textiles”

MIF37

Judith MILLET

## Sommaire :

# 1. Explication du système masses-ressorts

Un système masses-ressorts est un objet discrétisé en un ensemble de masses connectées par des ressorts qui vont permettre de déformer l'objet. Dans ce TP, l'objet qu'on souhaite déformer est un textile. On discrétise donc celui-ci en un maillage polygonal. Ainsi, pour le modéliser et l'animer, on peut émettre l'hypothèse qu'un tissu est un ensemble de particules qui ont une masse reliées par des ressorts (cf. Figure 1).

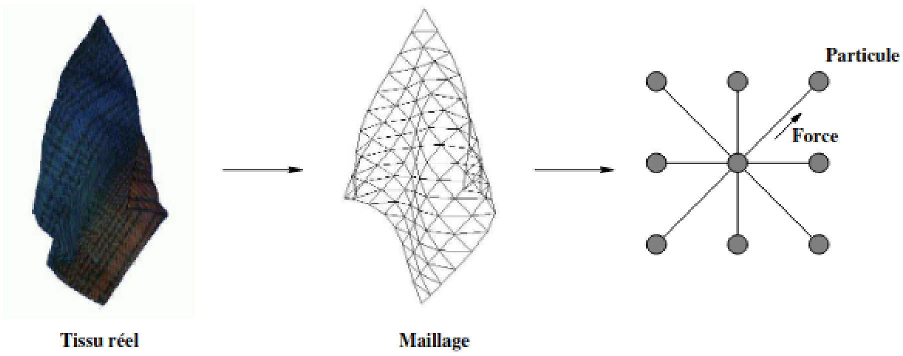


*Figure 1 : Schéma d'un tissu exprimé en un maillage polygonal.*

Les sommets du maillage sont les particules et les interactions sont représentées par des ressorts. Ce TP consiste donc à animer un tissu selon le système masses-ressorts.

Les étapes pour réaliser cette simulation sont les suivantes :
- Afficher l'objet simulé (le maillage)
- Calculer pour chaque particule :
    - la force exercée
    - l'accélération
    - la vitesse
    - la position

On pourra ensuite appliquer une interaction avec l'utilisateur, une possibilité de déchirure et des objets qui seront en collision avec le tissu.

# 2. Affichage de l'objet simulé

Pour afficher l'objet, il faut initialiser l'objet en un maillage de triangles. Nous avons à disposition une tableau déjà initialisé contenant les positions des particules, ainsi qu'un tableau contenant les normales pour chaque particule. Ainsi, pour chaque particule i, on peut ajouter un vertex à la position i ainsi que sa normale associée. Il est possible d'ajouter une texture à ce moment en ajoutant les coordonnées de la texture d'après un tableau initialisé avec les coordonnées de la texture.

Il faudra ensuite créer l'interconnexion entre les sommets en générant les triangles d'après un tableau contenant les indices des sommets du maillage. L'ordre des indices des sommets au sein de cette structure suit l'ordre des triangles à créer.

# 3. Calculs

## 3.1 Calcul des forces

Sachant que les particules sont connectées entre elles par des ressorts, on peut obtenir les forces appliquées sur les particules en calculant les forces exercées par les ressorts. En effet, lorsqu'une force est appliquée à une particule, le ressort est compressé ou étiré et il compensera en appliquant une force à la particule opposée pour décompresser ou se recompresser.

Le calcul d'une force $\vec{F}$ se fait dans la méthode *CalculForceSpring()* de la classe *ObjetSimuleMSS* grâce à la formule :

$$\vec{F} = \sum_{j|(i,j)\in E}\left[\vec{f}^{e}_{i,j}(t) + \vec{f}^{v}_{i,j}(t)\right]$$

$$\begin{cases} \vec{f}^{e}_{i,j}(t) & = & -k_{ij}\left(\|x_i(t) - x_j(t)\| - l_{ij}\right)\ \vec{u}_{i,j} \\ \vec{f}^{v}_{i,j}(t) & = & \left(-\nu_{ij}\left(\dot{x}_j(t) - \dot{x}_i(t)\right)\cdot \vec{u}_{i,j}\right)\vec{u}_{i,j} \end{cases}$$

Avec :

- k : la raideur du ressort,
- $\| x_i(t) - x_j(t) \|$ : la longueur actuelle du ressort,
- $l_{ij}$ : la longueur du ressort au repos,
- v : le coefficient d'amortissement,
- $\dot{x}_j(t) - \dot{x}_i(t)$ : la différence entre les vitesses des deux particules,
- $\vec{u}_{ij}$ (t) : vecteur normalisé allant de *i* vers *j*.

On pourrait ajouter à la force la force gravité et des forces externes à ce moment mais elles seront appliquées lors du calcul de l'accélération.

Ainsi, pour chaque ressort du système, on calcule la force exercée qui sera ajoutée à la force appliquée à la particule A et l'inverse sera appliquée la particule B.

## 3.2 Calcul des accélérations

En utilisant la loi fondamentale de la dynamique, on peut calculer l'accélération pour chaque particule en divisant toutes les forces appliquées à cette particule par sa masse. C'est à ce moment que la force de gravité est appliquée à chaque particule.

L'accélération est gérée dans la méthode *CalculAccel_ForceGravite()* de la classe *SolveurExpl*. Pour calculer l'accélération de chaque particule, on vérifie d'abord que sa masse est différente de 0 pour éviter une erreur de calcul. En effet, certaines particules peuvent avoir une masse égale 0 afin de les garder fixes. On applique ensuite à l'accélération $\vec{A}$ la formule :

$$\vec{A}_i = ( \vec{F}_i + \vec{g} ) / M_i.$$

Avec :

$\vec{F}_i$ : la force appliquée à la particule *i*,
$\vec{g}$ : la force de gravité,
$M_i$ : la masse de la particule *i*.

C'est dans cette méthode que le vecteur des forces est réinitialisé afin de pouvoir les recalculer à chaque pas de temps.

Lorsque la masse est égale à 0, on peut choisir de garder les coins fixes en appliquant une accélération nulle, ou changer sa masse en 1 afin de ne plus avoir de coins fixes et ainsi, laisser tomber le tissu.

## 3.3 Calcul des vitesses et positions

Au cours de la simulation, les vertex seront mis à jour en fonction des nouvelles positions. On implémente donc le schéma d'Euler explicite afin de calculer les nouvelles vitesses et positions.

Ces calculs sont effectués dans la méthode *Solve()* de la classe *SolveurExpl*. Ainsi, pour chaque particule *i*, on calcule sa vitesse $\vec{V}$ en fonction de son accélération et sa position $\vec{P}$ en fonction de sa vitesse, à un instant $t + \delta t$, avec les formules :

$$\vec{V}_i (t + \delta t) = \vec{V}_i (t) + \delta t * \vec{A}_i (t).$$
$$\vec{P}_i (t + \delta t) = \vec{P}_i (t) + \delta t * \vec{V}_i (t + \delta t).$$

Avec :

$\vec{V}_i$ (t) : sa vitesse à l'instant t,
$\vec{P}_i$ (t) : sa position à l'instant t,
$\vec{A}_i$ (t) : son accélération à l'instant t,
$\delta t$ : un pas de temps.

Afin de dissiper les vitesses, on multiplie les vitesses et les positions par le coefficient de viscosité qui est lié à la scène.

# 4. Interaction avec l'utilisateur (mouvement du coin)

On sait que les particules fixées sont celles placées aux coins haut-gauche et haut-droit. De plus, on sait que l'identifiant de celle placée en haut à gauche est 0. On peut donc modifier la position de cette particule particulièrement afin de créer une interaction. L'interaction est effectuée lorsque la touche 'm' est enfoncée et que l'une des flèches de direction est activée. Cette combinaison va permettre de déplacer le coin d'identifiant 0 initialement fixe du tissu.

Ainsi, la méthode *Interaction(Vector MousePos)* de la classe *ObjetSimule* prend comme paramètre un vecteur correspondant au changement à ajouter dans la position.

Si on appuie sur la touche →, le vecteur en paramètre est : V(0.01, 0, 0),
Si on appuie sur la touche ←, le vecteur en paramètre est : V(-0.01, 0, 0),
Si on appuie sur la touche ↑, le vecteur en paramètre est : V(0, 0, -0.01),
Si on appuie sur la touche ↓, le vecteur en paramètre est : V(0, 0, 0.01).

Ce paramètre est donc ajouté à la position actuelle de la particule fixe. Cette nouvelle position est modifiée dans le maillage du tissu à l'identifiant 0. Le point P[0] est aussi mis à jour avec cette nouvelle position.

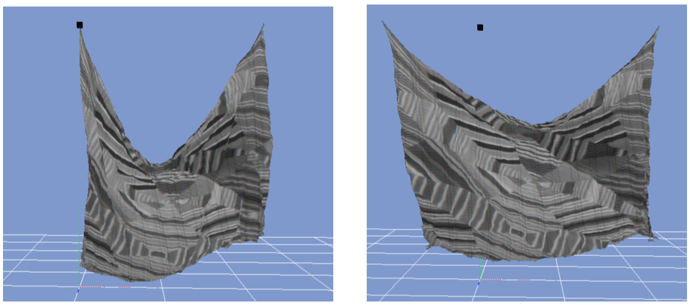

*Figure 2 : Drap avant interaction (à gauche) et après interaction (à droite).*

# 5. Déchirure

La déchirure est gérée au moment du calcul des forces appliquées aux particules. Ainsi, pour chaque ressort, si la distance entre les 2 particules aux extrémités est supérieure à un seuil, on supprime le ressort pour créer la déchirure. De ce fait, il n'existe plus de ressort qui exerce une force sur les deux particules, ce qui déconnecte les deux particules par conséquent. Le seuil choisi est 0.7.

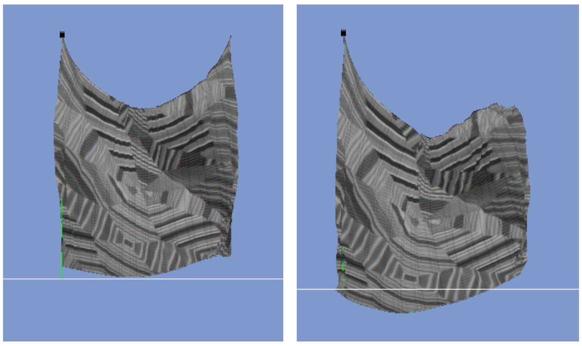

*Figure 3 : Drap avant déchirure (à gauche) et après déchirure (à droite).*

## 6. Collisions

Les collisions sont détectées et traitées dans la méthode *Collision()* de la classe *CalculsMSS*. Ces collisions sont effectuées avec des objets dont on connaît l'emplacement afin de faciliter leur détection.

En effet, on sait que le sol est un plan horizontal de position P(0, 0, 0) et de normale N(0, 1, 0). Une particule est donc en collision avec le sol lorsque sa position en y, passé dans le repère de la scène, est inférieure ou égale à 0. Dans ce cas, la particule est au niveau du sol et on met sa vitesse à 0 afin de la stopper.

De plus, on peut placer une sphère dans la scène qui peut engendrer une collision avec le tissu. Pour se faire, on récupère la position de la sphère et son rayon qui est égale à la distance entre le point minimum et le point maximum du maillage de la sphère divisée par deux. On peut ensuite vérifier si la distance entre la position d'une particule et le centre de la sphère est inférieure ou égale au rayon de la sphère. Dans ce cas, il y a une collision entre la particule et la sphère. On peut élaborer de la même manière qu'avec le plan en mettant la vitesse de la particule en question à 0. Pour un peu plus de réalisme, on peut simplement mettre la composante y de la vitesse à 0 afin de simuler un tissu qui glisse sur la sphère.

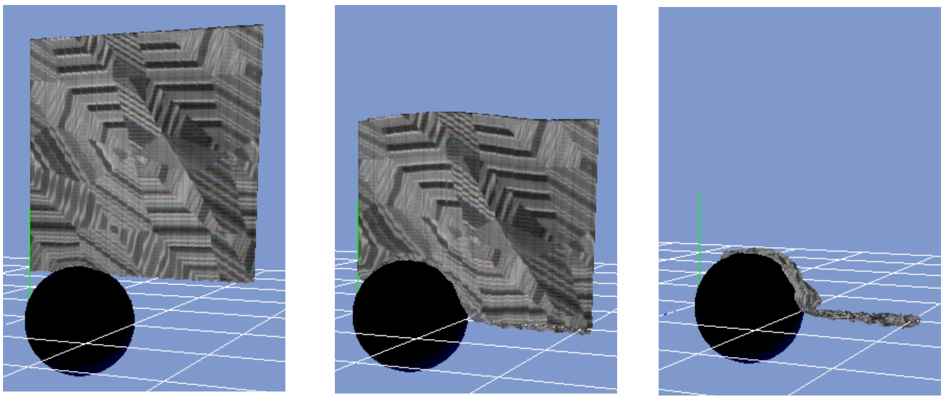

*Figure 4 : Drap subissant une collision avec une sphère et avec le sol à plusieurs instants.*

# 7. Perspectives

Si j’avais plus d’heures, je modifierais la gestion des collisions avec les objets, en ajoutant une force perpendiculaire à la norme de l’objet détecté. Les collisions actuelles avec la sphère ne modifient que la vitesse en $y$ afin de mieux simuler la collision. Par ailleurs, il faudrait aussi gérer les collisions avec un cube axé dans un premier temps en détectant la face sur laquelle il y a une collision.

De plus, je créerais une scène plus aboutie contenant plus de décors et d'interactions. La scène pourrait prendre en compte un nouvelle force externe telle que le vent et contenir plusieurs cubes lorsque la gestion des collisions avec ceux-ci sera établie.