TP Synthèse d'images: Simulation physique
- Animation de tissus -
CPE

durée - 4h

2011-2012

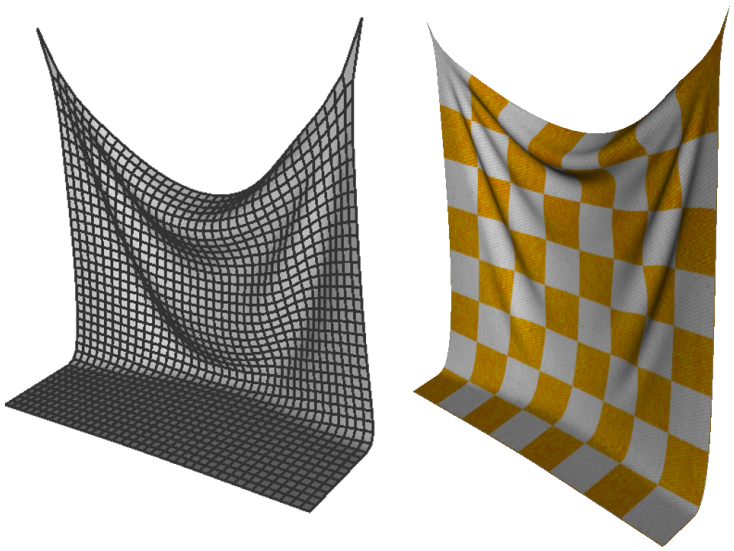


FIGURE 1 – Resultats possibles de la simulation de tissus.

1 But du TP

Le but de ce TP est d'implémenter le calcul et l'integration temporelle des forces de ressorts modélisant une simulation de tissus (voir fig. 1). L'integration se réalisera par la méthode de explicite d'Euler.

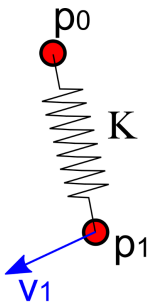
2 Prise en main de l'environnement

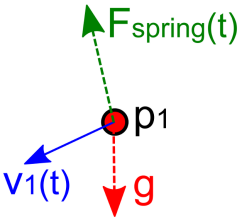
Le chargement de la scène et son affichage sont donnés dans la classe *scene*.

La simulation et l'affichage d'un ressort 3D est effectuée dans cette classe (voir fig. 2). La méthode *load model* est appelée une unique fois au lancement du programme. Elle sert donc à donner les valeurs initiales de la simulation. La méthode *draw scene* est appelée en permanence pour un affichage en teps-réel. On y effectue l'affichage, ainsi que la simulation (suivant une fréquence controlée).

Question 1 *Observez et comprennez l'articulation globale de ce programme.*

```
//cette méthode est appelée une unique fois en début de programme
void scene::load_model()
{

    // ********************************* //
    // position initiale du ressort
    // ********************************* //
    p0=cpe::v3(0.1,0,1);   // position fixe
    p1=cpe::v3(0.1,0,0.5); // position du bas du ressort
    v1=cpe::v3(1,-1,0);    // vitesse initiale de la position p1
    L0=(p1-p0).norm();     // longueur au repos du ressort

}

//Cette méthode est appelee en permanence pour l'affichage
void scene::draw_scene()
{
    glEnable(GL_LIGHTING);
    glEnable(GL_TEXTURE_2D);

    // ********************************* //
    // Calcul des forces
    // ********************************* //
    cpe::v3 gravity(0,0,-9.8); //gravitee
    cpe::v3 u=p1-p0; //vecteur "ressort"
    double L=u.norm(); //longueur du ressort
    cpe::v3 force=K * (L0-L) * u/L + gravity; //force du ressort



    // ********************************* //
    // integration temporelle des forces
    // ********************************* //
    v1 = (1.0-mu*dt)*v1+dt*force;
    p1 = p1 + dt*v1;



    //Le reste concerne l'affichage
    //...
}
```





$v_1(t+dt)=v_1(t)+dt\ F(t)$

$p_1(t+dt)=p_1(t)+dt\ v_1(t)$

FIGURE 2 – Scene de simulation et d'affichage du ressort.

3 Simulation d'un ressort

3.1 Ressort unidimentionnel

On rappel qu'en 1D, un ressort est modélisé par une force de rappel F s'appliquant en une position p de coordonnée x tel que :

$$F(t) = K\left(L_0 - (x(t) - x_0)\right). \tag{1}$$

- K est appelé la constante de raideur du ressort.
- L_0 est la longueur au repos du ressort.
- x_0 étant l'autre extremitée du ressort.

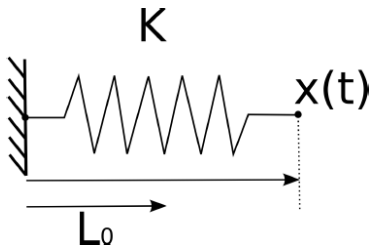


FIGURE 3 – Modelisation d'un ressort unidimentionnel.

Question 2 *Rappelez l'équation du mouvement d'un tel ressort dans le cas d'une simulation en temps continu.*

3.2 Ressort 3D

Soit le ressort 3D liant les positions $\mathbf{p}_0$, $\mathbf{p}_1$. On supposera $\mathbf{p}_0$ fixe, et on s'interesse à $\mathbf{p}_1$. La position de coordonnées $\mathbf{p}_1 = (x_1, y_1, z_1)$ est cette fois soumise à la force de rappel du ressort s'écrivant

vectoriellement :

$$\mathbf{F}(t) = K\ (L_0 - \|\mathbf{p}_1 - \mathbf{p}_0\|)\ \frac{\mathbf{p}_1 - \mathbf{p}_0}{\|\mathbf{p}_1 - \mathbf{p}_0\|}\ . \tag{2}$$

3.3 Simulation complète

Pour une simulation complète, il est possible d'ajouter la force de gravité $\mathbf{g}$. On peut également considérer une force de frottement fluide (*damping*) tel que

$$\mathbf{F}_d = -\mu\mathbf{v}\ , \tag{3}$$

où $\mathbf{v}$ est la vitesse du point $\mathbf{p}$, et μ est le coefficient d'amortissement.

L'équation du mouvement est donc donné par

$$\left\{\begin{array}{ll} m\,\dfrac{\partial \mathbf{v}}{\partial t}(t) & = \mathbf{F}(t) + \mathbf{F}_d(t) + \mathbf{g} \\ \dfrac{\partial \mathbf{p}}{\partial t}(t) & = \mathbf{v}(t)\ , \end{array}\right. \tag{4}$$

où m représente la masse de la position.

3.4 Simulation numérique

La version discrétisée de l'équation précédente suivant la méthode dite *d'Euler explicite* consiste à considérer l'approximation

$$\left\{\begin{array}{ll} m\,\dfrac{\mathbf{v}(t+\Delta t) - \mathbf{v}(t)}{\Delta t} & = \mathbf{F}(t) + \mathbf{F}_d(t) + \mathbf{g} \\ \dfrac{\mathbf{p}(t+\Delta t) - \mathbf{p}(t)}{\Delta t} & = \mathbf{v}(t)\ , \end{array}\right. \tag{5}$$

On obtient ainsi la relation réccursive implémentée dans le programme de démonstration en considérant une masse $m = 1$:

$$\left\{\begin{array}{l} \mathbf{v}(t+\Delta t) = (1 - \mu\Delta t)\,\mathbf{v}(t) + \Delta t\,(\mathbf{F}(t) + \mathbf{g}) \\ \mathbf{p}(t+\Delta t) = \mathbf{p}(t) + \Delta t\,\mathbf{v}(t) \end{array}\right. \tag{6}$$

Question 3 *Identifiez la partie d'intégration numérique dans la classe* scene. *Prenez soin de différencier la partie* initialisation, *la partie de* calcul des forces, *la partie d'*intégration numérique *de la simulation, et enfin, l'*affichage.

Question 4 *Observez le comportement de la simulation en modifiant les paramètres de : pas de simulation Δt, coefficient d'amortissement, constante de raideur K. Que ce passe-il lorsque K est trop grand ? Δt grand ? Est-ce attendue de part l'étude du comportement d'un ressort à temps continu ? Expliquez ce phénomène.*

3.5 Ressorts couplés

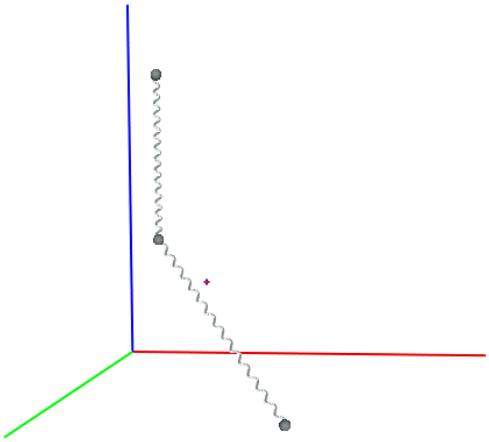


FIGURE 4 – Exemple de ressorts couplés.

Question 5 *Ajoutez un ressort supplémentaire, pour obtenir une simulation de ressorts couplés (voir fig. 4).*

4 Simulation de tissu

Dans le reste du sujet, nous allons généraliser le couplage de ressorts afin de modéliser une surface maillée. Chaque sommet du maillage est alors relié à ces voisins par des ressorts (voir fig. 5).

Simuler un tel système permet alors de modéliser le comportements d'objets déformables. En particulier, on va considérer le cas d'une simulation de tissu.

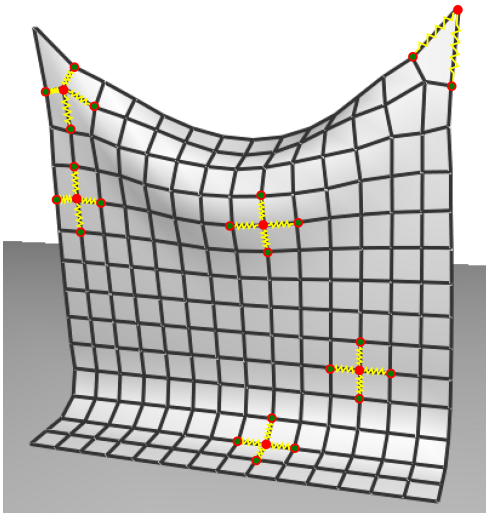


FIGURE 5 – Exemple de surface déformable formée par un ensemble de ressorts (seulement quelques ressorts sont affichés).

4.1 Classe Spring Mesh

Une nouvelle classe apparait : *spring mesh*. Il s'agit de la classe de gestion des ressorts liées.
Son fonctionnement est le suivant :

- *constructeur* : Construit une structure rectangulaire de N×N sommets
- *compute force* : Calcule les forces de gravité + des ressorts et les stocke dans le conteneur *forces*. Cette methode est à compléter.
- *apply force* : Realise l'intégration temporelle des forces ainsi que de la vitesse sur les positions. L'intégration peut se réaliser suivant Euler explicite, et le pas de temps est passé en paramètre.

Cette classe permet egalement de gérer 3 raideurs de ressorts différentes.

4.2 Fonctionnement général

Un premier et unique passage dans la fonction *load model* est réalisé en début de programme.
Ensuite la fonction *draw scene* est appelé en permanence au cours du déroulement. C'est dans celle-ci que l'on réalise la MAJ des forces et l'integration temporelle.

4.3 Intégration temporelle

L'intégration suivant la méthode d'Euler explicite suit l'algorithme suivant

```
for every vertices k
  speed[k] = (1-dt*damping) speed[k] + dt forces[k]

for every vertices k
  vertices[k] += dt speed[k]
```

On notera que dans cette implémentation les vecteurs sont concaténés suivant x, y, z. Voici quelques exemples d'accès aux éléments :

```
forces[3*k+1];   // coordonnée y de la force du sommet k
speed[3*2+0];    // coordonnée x de la vitesse du second sommet
vertices[3*i+2]; // coordonnée z du sommet i
vertices[3*((kx-1)+N*(ky+1))+1]; // coordonnée y du sommet
                                 // (kx-1,ky+1) dans la grille 2D
```

Question 6 *Complétez la méthode* apply force *afin de réaliser l'integration temporelle suivant la méthode d'Euler explicite.*

On notera que l'on peut forcer la valeur de la vitesse ou de coordonnées pour certains sommets dans cette méthode. Ainsi si l'on souhaite fixer entièrement le sommet k, il suffit de ne jamais mettre à jour ses coordonnées.

De même, en projettant la vitesse suivant un axe/plan donné, on pourra contraindre artificiellement le mouvement des sommets suivant ce degré de liberté.

Question 7 *Fixez deux sommets des coins du tissu.*

4.4 Mise en place des forces

La MAJ régulière des forces se réalise dans la méthode *compute force*. Pour l'instant seul une force de gravité est appliquée. Le tissu doit donc tomber avec une accélération constante.
Il convient de calculer les forces de rappels des ressorts et de mettre à jour le vecteur de forces pour chaque sommet. On désigne par s(ku,kv) le sommet s paramétré par (ku, kv).
On pourra séparer les ressorts suivant 3 types (voir fig. 6) :

- Ressorts structurels liant le sommet courant s(ku,kv) à
  - s(ku+1,kv)
  - s(ku,ky+1)
  - s(ku-1,kv)
  - s(ku,kv-1)

- Ressorts de cisaillement (shear) liant le sommet courant s(ku,kv) à
  - s(ku+1,kv+1)
  - s(ku-1,kv+1)
  - s(ku-1,kv-1)
  - s(ku+1,kv-1)
- Ressorts de courbure (bending) liant le sommet courant s(ku,kv) à
  - s(ku+2,kv)
  - s(ku,kv+2)
  - s(ku-2,kv)
  - s(ku,kv-2)

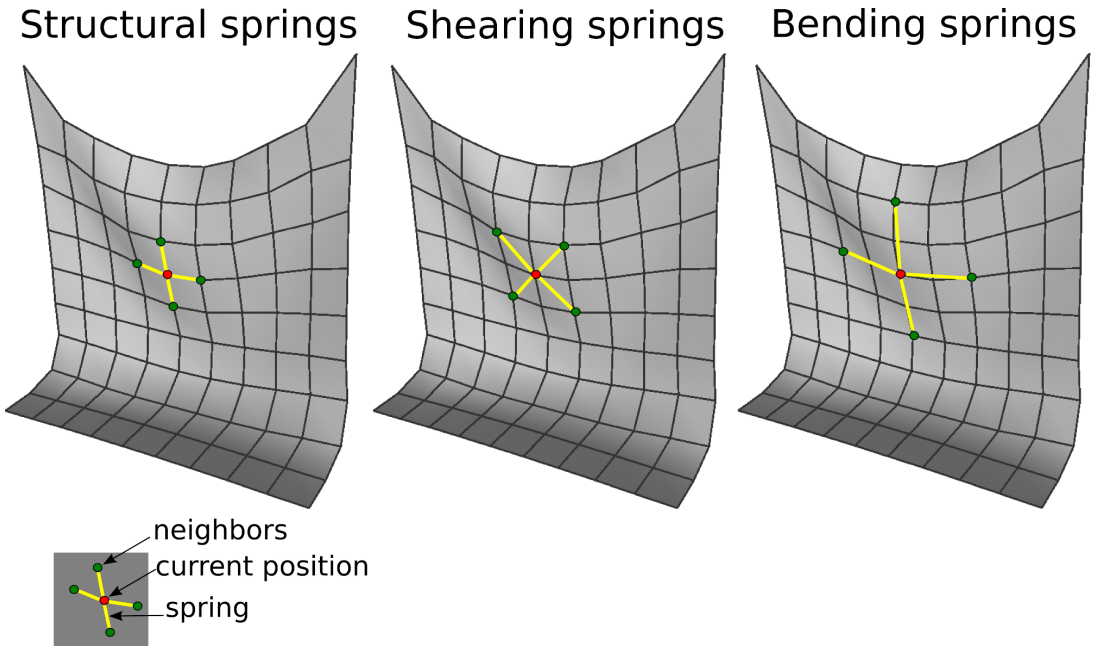


FIGURE 6 – Les trois types de ressorts appliqué au point courant (cercle rouge).

Ces 3 types de ressorts peuvent avoir des raideurs différentes en fonction du type de tissu simulé.

On notera que l'on pourra récupérer le vecteur de coordonnées du sommet s(ku,kv) par la syntaxe suivante :

```
unsigned int u=3*(ku+N*kv);
v3 x_current=v3(vertices[u+0],vertices[u+1],vertices[u+2]);
```

Question 8 *Complétez la méthode* compute force *de manière à calculer les forces des ressorts.*

4.5 Etude complémentaire

Question 9 *Modélisez l'action d'un plan d'équation $z = -1$: Les sommets sont contraints à rester sur ce plan si $z < -1$ et les sommets en contacts pourront subire un frottement de type fluide.*

Question 10 *Observez l'influence de la subdivision du maillage sur la raideur. Augmentez la raideur et commentez son influence sur la stabilité du système. Que faut-il faire pour rendre le système plus stable ? Commentez.*

5 Implémentation sur GPU

Le dernier programme implémente une déformation calculée sur GPU. La position, vitesse et les normales des sommets sont passées en tant que texture. Trois shaders différents viennent remplir leurs valeurs.

Question 11 *Observez la structure du programme et différenciez la génération des textures liées aux positions, aux vitesses, aux normales et aux vraies coordonnées de textures.*

Quel est le type de shader utilisé ? Quelle est la géométrie 3D d'entrée envoyée à la carte graphique par le CPU ?

Question 12 *Complétez le shader de mise à jour de la vitesse en ajoutant les forces de rappels des ressorts. Comparez la vitesse d'execution par rapport à la version CPU précédente.*

La force exercée par le vent sur un tissu peut être modélisée par une force agissant dans la direction normale à la surface, et proportionnelle à l'angle entre la normale et la direction du vent. Pour cela, on peut considérer une force F_w telle que

$$F_w = K_w \ < \mathbf{n}, \mathbf{u}_w > \ \mathbf{n} \ ,$$

où K_w est une constante correspondant à l'intensité du vent, $\mathbf{n}$ est la normale à la surface, et $\mathbf{u}_w$ la direction du vent.

Question 13 *Ajoutez la force du vent dans votre simulation.*