TP 5 et 6 d'OpenGL
Animation par modèles physiques

Dans ce (double) TP, nous allons implémenter deux modèles d'animation physique: les systèmes masse-ressort et les systèmes de collision. Aidez-vous de la feuille de cours sur les systèmes dynamiques (fichier systDyn.pdf sur la page du Kiosk) pour comprendre le code fourni ainsi que le travail à accomplir.

1. Schéma d'intégration

Recuperez le code source fourni (fichier progTP_5_6.zip sur la page du Kiosk), compilez (avec make) puis exécutez. Le programme représente une boule qui se translate suivant sa vitesse initiale. Il y a aussi une boule fixe, dont on ne s'occupe pas dans un premier temps.

Observez tout d'abord le fichier prog.c : il comporte les méthodes vues précédement pour prendre en compte la camera (mêmes touches du clavier), pour calculer un pas de temps et pour dessiner une sphère et un plan. De nouvelles méthodes sont ajoutées : elles utilisent un objet de la classe dynamics qui va nous permettre d'implémenter des modèles d'animation physique.

En dehors de la boule fixe qui peut être contrôlée à la souris (nouvelles méthodes mouseClick() et mouseMove()) et du plan qui va rester fixe, toute animation va être contrôlée par l'intermédiaire de l'objet dynamics: à chaque pas de temps, on appelle la fonction d'animation de dynamics (cf. updateAnimationParameters()) qui va positionner nos objets ; puis on dessine les boules et ressorts en recuperant les informations de position calculées par dynamics (cf. drawBalls() et drawSprings()).

Examinez dans le détail les fichiers dynamics.h et dynamics.c, en particulier le calcul des forces et le schéma d'intégration utilisé. La pesanteur y est prise en compte, et pourtant la vitesse de la boule reste constante lors de l'exécution.

Question 1 Complétez le schéma d'intégration en mettant à jour la vitesse en fonction des accélérations afin de tenir compte de la pesanteur dans l'animation. Observez le résultat à l'exécution.

Question 2 La boule fixe est un objet dynamique au même titre que la boule mobile. En analysant le programme, expliquez pourquoi elle est fixe.

2. Modèle masse-ressort

Nous allons maintenent construire progressivement une chaine de boules liées deux a deux par un ressort amorti, que l'on pourra manipuler interactivement.

Question 3 Dans la méthode setDynamics(), ajoutez un ressort entre la boule fixe et la boule mobile à l'aide de la méthode addSpring() de Dynamics. Utilisez les variables fournies pour les paramètres des ressorts. Observez les résultats sur le rendu et l'animation qui réagit aux évènements souris en déplaçant la boule «fixe».

Question 4 Ceci fait, modélisez une chaîne de boules accrochée en une extrémité à la boule fixe, ayant mêmes masses, rayons, et paramètres de ressorts. Le nombre de boules créées devant être paramétré par la variable *nbBalls*. Expérimentez avec les valeurs courantes, puis changez les valeurs et observez les réactions.

Question 5 Afin de dissiper les vitesses, modélisez une force d'amortissement visqueux (voir feuille de cours) agissant sur chaque boule. Visuellement, quelle est la différence d'effet entre le coefficient de dissipation qu'on vient d'introduire et le coefficient d'amortissement des ressorts ?

3. Modèles de collision

Notre chaine réagit de manière convaincante, mais les boules traversent le sol et s'inter-pénètrent fréquement. Nous allons ajouter maintenant des collisions pour obtenir un notre système final.

Question 6 Décommentez les lignes nécessaires pour traiter les collisions entre les boules et le sol. Expérimentez en manipulant la boule fixe lors de l'exécution.

Question 7 En vous inspirant très fortement de l'implémentation boule-plan, traitez la collision boule-boule dont la méthode (vide) est fournie.

Vous disposez maintenant d'un système complet. Continuez à expérimenter en modifiant des paramètres physiques, ou en essayant de placer le système dans des positions d'équilibre (ex: toutes les boules les unes au dessus des autres à la verticale en position initiale).