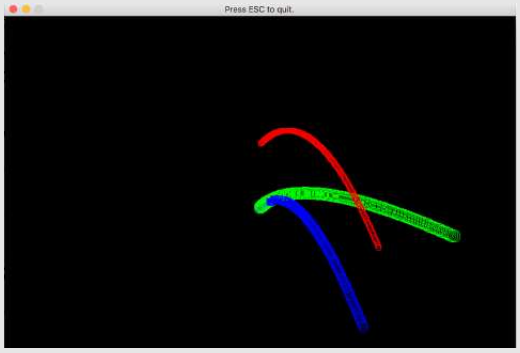
IN 103 - TD 03

Énoncé

ENSTA ParisTech

Dans ce TD, on souhaite développer un petit moteur de physique simple en 2 dimensions. Le moteur doit être capable de simuler le mouvement de corps donnés sur lesquels s'appliquent des forces également données.

L'affichage donnera lieu à une animation des mouvements. L'image ci-dessous montre un résultat attendu, en ayant laissé la trace des positions successives pour le besoin de l'illustration statique.



1 Les corps

Q1 Sachant qu'un corps est caractérisé par :

- sa couleur de dessin,
- le rayon du cercle qui le représente à l'écran,
- sa masse,
- ses coordonnées x et y,
- sa vitesse en x et en y,
- son accélération en x et en y,

déduisez-en une structure de données pour le représenter.

Q2 Sachant que l'on souhaite pouvoir rajouter ou supprimer dynamiquement des corps dans la simulation, proposez une structure de données pour mémoriser l'ensemble des corps d'une simulation.

2 Les forces

Q3 On souhaite permettre l'appliction de forces non constantes, possiblement dépendantes de la position, de la vitesse du corps sur lequel elles s'appliquent. Proposez une représentation d'une force.

Q4 Vu qu'une force est caractérisée en 2 dimensions par ses composantes x et y, comment pouvez-vous retourner ces deux composantes aux fonctions qui ont besoin de travailler avec une force ?

Bien entendu, la réponse de **Q4** nous indique que nous définirons nos forces statiquement. Nous allons les stocker un tableau statique terminé par `NULL` (qui nous servira pour savoir quand nous aurons atteint la fin lors des parcours.

Dans le fichiers `body.h`, `draw.h` et `simu.h` vous sont données les structures de données que les questions précédentes vous invitaient à concevoir.

En outre se trouvent les prototypes des différentes définitions que vous devrez écrire. Utilisez donc ces fichiers d'en-tête pour la suite.

3 Structure de l'algorithme de simulation

Le moteur de simulation va donc appliquer les lois de la physique newtonienne en boucle, chaque itération de la boucle faisant avancer le ≪temps≫ (et affichant la position des corps). Il convient donc de définir un ≪delta de temps≫ qui représentera le temps physique écoulé entre deux étapes successives de la simulation. Dans la suite nous le nommerons `DT`. Sa valeur sera à votre discrétion.

Q5 Puisque notre objectif est de simuler le mouvement de corps soumis à des forces, quelle va être la forme de l'algorithme ?

4 Manipulation des corps

Q6 Écrivez la fonction `new_body` qui crée un nouveau corps et l'ajoute à la structure de mémorisation déduite en question **Q2**.

Q7 Quelles sont les équations qui permettent de calculer le nouvelle vitesse et la nouvelle position d'un corps ?

Q8 Écrivez la fonction `move_body` qui met à jour la vitesse et la position du corps qu'elle reçoit en argument.

Désormais, nous savons mettre à jour la vitesse et la position d'un corps à partir de l'accélération qu'il subit. Il nous faut donc calculer cette accélération à partir des forces appliquées.

5 Manipulation des forces et simulation

Q9 Définissez autant de forces que vous souhaitez et stockez-les dans un tableau.

Q10 Quelles sont les équations qui régissent l'accélération subie par un corps en fonction de la force qu'il subit ?

Q11 Écrivez la fonction `apply_forces_on_body` qui prend en argument un corps, les forces et applique chacune des forces sur ce corps pour mettre à jour l'accélération de ce corps.

Q12 Et la gravité ? Comment pouvons nous intégrer la gravité dans notre moteur ?

Q13 Maintenant que nous savons intégrer la gravité dans la simulation, écrivez une fonction `simulate_bodies` qui prend en arguments les corps et les forces et applique les forces et met à jour la position de chaque corps.

Les fichiers `draw.(c|h)` et `physics` vous sont donnés. Le premier contient une fonction `draw_bodies` permettant d'afficher (ou d'effacer) tous les corps, le second contient le `main`. Le `main` s'attend à disposer de la fonction `simulate_bodies` que vous avez écrite à la question précédente.

Q14 Compilez votre programme et testez-le. Pour compiler, vous devrez utiliser la ligne de commande ci-dessous, en rajoutant les fichiers `.c` que vous aurez écrits à ceux qui vous sont fournis.

```
gcc gfxprims.c VOS_FICHIERS draw.c physics.c ‘sdl-config --cflags --libs‘
```