N-Body Problem

Hugo MANGNAN
Antoine JOURDAN
Kilian CAILLOT
Paul GOSSE

Mai 2020

Table des matières

Introduction

Énoncé du sujet choisi :

Le problème à N corps est un problème d'astronomie classique où plusieurs corps se déplacent dans l'espace en étant soumis à leur propre inertie et l'attraction des autres corps. L'équation différentielle qui modélise ce problème est en pratique inutilisable pour $N > 2$. Le but de ce projet est dans un premier de simuler un espace newtonien où N corps interagissent et visualiser cette simulation. Il s'agira ensuite d'améliorer ce simulateur avecdiverses propositions parmi les suivantes : instancier des chorégraphies à N corps, accélérer l'optimisation avec un découpage spatial récursif, intégrer un jeu de pilotage d'un corps au clavier, développer une IA pour optimiser les déplacements avec une trajectoire faible en énergie comme les orbites de transfert.

Théorie

Implémentation algorithmique

- La méthode Leapfrog : elle permet la résolution d'équations différentielles du second ordre par une méthode de récurrence.

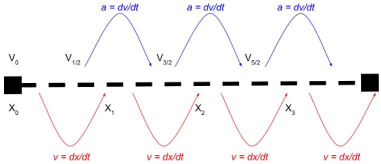


Figure 1 – Schéma de la méthode de Leapfrog

- Les méthodes Runge-Kutta : elles vont notamment composer la méthode d'Euler.

Théorie

- La méthode d'Euler : elle permet la résolution d'équations différentielles du premier ordre.

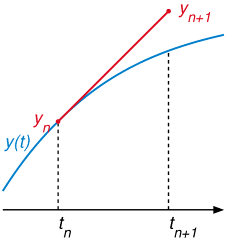


Figure 2 – Schéma de la méthode d'Euler

L'avancée se fait par approximation sur la tangente au point initial.

Architecture

Classe du modèle :

- La classe **Corps**
- La classe **Point**
- La classe **Système**
- La classe **Vecteur**

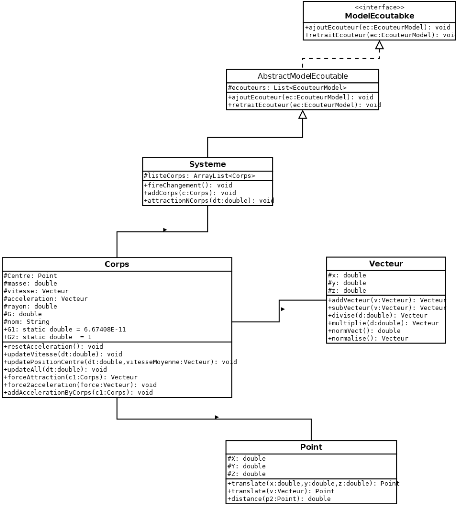


Figure 3 – Diagramme du modèle

Éléments techniques

Implémentation mathématique

$$\overrightarrow{F} = G \times \frac{m1 \times m2}{R^3} \times \overrightarrow{R}$$

- $\overrightarrow{F}$: vecteur Force
- G : constante gravitationnelle
- m1 : masse du corps 1
- m2 : masse du corps 2
- R : distance entre les deux corps
- $\overrightarrow{R}$: vecteur entre le corps attiré vers le corps attracteur

Figure 4 – Calcul du vecteur Force

Éléments techniques

Implémentation des données

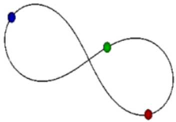
Corps	Coordonnées initiales (x,y,z)	Masse (kg)	Rayon (km)	Vecteur vitesse initiale (x,y,z)
Soleil	0 0 0	1.98855E+30	639934	0 0 0
Mercure	1.40191E+10 -6.64159E+10 -6.71319E+9	0.33011E+24	2440	3.78936E+ 1.25295E+01 -2.45039E+01
Venus	-9.90252E+10 4.14922E+10 6.28376E+9	4.8675E+24	6052	-1.36922E+01 -3.24633E+01 3.44688E-01
Terre	-1.44286E+11 -3.97378E+10 1.86460E+6	5.9723E+24	6371	7.42866E+00 -2.88200E+01 7.64891E-04
Mars	-2.68704E+10 -2.18308E+11 -3.91509E+9	0.64171E+24	3390	2.49637E+01 -8.76675E-01 -6.30821E-01
Jupiter	1.83152E+11 -7.54925E+11 -9.62242E+8	1898.19E+24	69991	1.25518E+01 3.70149E+00 -2.96220E-01
Saturne	6.36389E+11 -1.35785E+12 -1.72278E+9	568.34E+24	58232	8.22566E+00 4.07725E+00 -3.98732E-01
Uranus	4.38375E+12 -9.08578E+11 -8.23305E+10	102.413E+24	24622	-4.04915E+00 5.18937E+00 7.16753E-02
Neptune	4.38375E+12 -9.08578E+11 -8.23305E+10	86.813E+24	25362	1.07933E+00 5.35931E+00 -1.34968E-01

Figure 5 – Données des planètes du système solaire

Éléments techniques

Application des chorégraphies

- Chorégraphie en 8 :



Données :
positions :
(x1,y1) = (-0.97000436, 0.24308753),
(x2,y2) = (-x1, -y1), (x3,y3) = (0,0)
vélocités :
(vx1,vy1) = (vx2, vy2) = -(vx3, vy3)/2 ;
dont (vx3,vy3) = (0.93240737, 0.86473146)

- Chorégraphie en moth :

Données :
positions :
(x1,y1) = (-1, 0),
(x2,y2) = (-x1, -y1), (x3,y3) = (0,0)
vélocités :
(vx1,vy1) = (vx2, vy2) = -(vx3, vy3)/2 ;
dont (vx3,vy3) = (0.464445, 0.396060)

Éléments techniques

Rendu graphique :

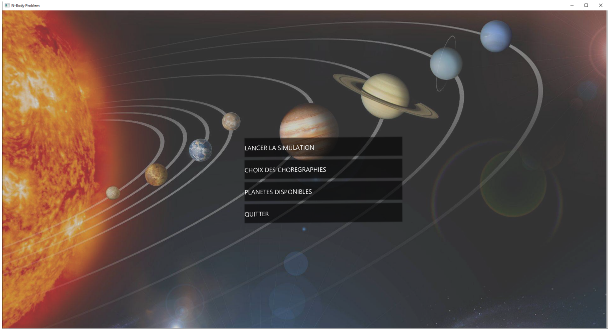


Figure 6 – Menu principal de notre application

Éléments techniques

Interface graphique :

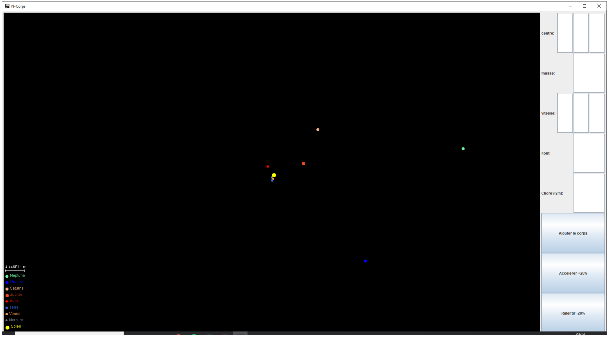


Figure 7 – Interface graphique de la simulation

Conclusion

Améliorations possibles :

- Optimisation des calculs et méthodes
- Meilleure gestion des scènes et fenêtres
- Un système de trajectoire pour suivre les corps
- Utilisation d'un découpage spatial récursif
- Intégrer un jeu de pilotage d'un corps au clavier
- Intégrer une IA pour optimiser les trajectoires