Département
du Premier
Cycle

Projet Informatique 2ème Année

Simulation mécanique d'un solide et moteur graphique 2D en Pascal

Tristan Pourcelot
Victor Cleren
Rémi Verschelde
Théo Garcia-Guitton

Responsable : C. Wolf

Groupe 52 Année scolaire 2010-2011

Institut National des Sciences Appliquées de Lyon

1 Introduction

Nous sommes partis d'une idée simple, celle de modéliser la chute d'un objet sur un décor, sachant que l'utilisateur pouvait influer à la fois sur le décor et sur l'objet. Nous avons également voulu intégrer de nombreux paramètres tels que la variation du matériau constituant le décor et l'objet, ainsi que la forme de cet objet. L'utilisateur pourrait ainsi (au prix de quelques approximations) simuler la chute d'un satellite sur la Lune.

2 Spécifications

L'objet principal de notre programme est de modéliser la chute d'un solide dans un fluide, tout en prenant en compte son interaction avec un environnement solide (chocs).

2.1 IHM - *Interface Homme - Machine*

Il est proposé à l'utilisateur de dessiner à la fois le décor et l'objet. Les deux dessins sont séparés en deux fenêtres, munie chacunes de leur interface propre, puisque les outils nécessaires au dessin du décor ne sont pas les mêmes que pour le dessin de l'objet. L'utilisateur peut également ouvrir une fenêtre dédiée aux paramètres physiques, afin d'influer sur la valeur de la gravité notamment.

2.2 Problèmes physiques et méthodes de résolution

Le programme doit prendre en compte les différentes forces appliquées à un instant t sur le solide. Le mouvement de ce solide est donc déterminé par les équations de la mécanique newtonienne. On peut donc définir les différentes forces appliquées sur le solide :

- La gravité : ..... $\overrightarrow{P} = -m * g * \overrightarrow{y_0}$
- La poussée d'Archimède : ..... $\overrightarrow{F_{Ar}} = -\rho_{liquide} * V_{deplace} * \overrightarrow{y_0}$
- Les frottements lors du contact : ..... $\overrightarrow{F_{fr}} = -f * \overrightarrow{V}_{\text{Objet/Décor}}$
- La force lors du choc : ..... $\|\overrightarrow{V_r}\| = e * \|\overrightarrow{V_i}\|$ avec $e \in [0, 1]$

La majorité des forces sont résolues à l'aide d'algorithmes basiques de calcul, prenant en compte des paramètres telles que le volume de l'objet, sa vitesse ou son poids. Les deux forces nécessitant de gros algorithmes de traitement sont les forces de frottement et le contact entre le solide et le décor. Pour la gestion du contact entre les deux solides, nous avons privilégiés une approche simplifiée, en considérant le point d'impact comme une surface plane telle que $\theta_r = -\theta_i$. De même, la vitesse du solide est donnée par la formule des chocs élastiques avec dissipation.

3 Réalisation

3.1 Interface

La fenêtre principale affiche un menu utilisateur, une fenêtre de dessin, et un espace d'affichage des principaux paramètres physiques. Outre ces fonctionnalités, c'est également à elle d'effectuer l'animation de l'image, en calculant régulièrement (présence d'un Timer) la position du solide ainsi que sa vitesse et son interaction avec le décor. Les autres fenêtres ont pour mission de gérer le dessin de l'objet et son remplissage, le dessin du décor et la gestion des matériaux. Nous avons également prévu une fenêtre permettant de changer les paramètres physiques.

3.2 Gestion des chocs

La partie la plus compliquée dans la gestion des chocs reste la détection des chocs et le calcul de la tangente locale au décor. Afin d'obtenir ces informations, notre algorithme analyse à chaque itération d'un timer prédéfini les pixels autour du solide, et si jamais ceux ci appartiennent à un des matériaux (tri par couleur), on calcule la tangente locale à l'aide de tout les points de contact. Ceci nous permet de déterminer la direction de rebond du solide, ainsi que d'appliquer les forces de frottement propres au matériau. Par ailleurs, nous appliquons la formule des chocs élastiques afin de calculer la nouvelle vitesse du solide.

3.3 Remplissage de l'objet

Un des problèmes majeurs qui nous fut posé durant ce projet fut le remplissage du solide afin de pouvoir déterminer sa masse et son centre d'inertie. Tout d'abord, il fallut s'assurer que le dessin de l'utilisateur était bien connexe. Pour remédier à ce problème, nous traçons une droite entre le point de départ (X_e, Y_e) et le point de sortie (X_s, Y_s). Nous avons choisi la méthode du ScanLine-FloodFill pour remplir le solide dessiné. Cette méthode utilise une graine (la *seed*) placée sur un point quelquonque de notre solide, et analyse tout les pixels environnants pour déterminer quels sont ceux à l'intérieur du solide, et les autres. Après plusieurs essais infructueux avec un algorithme classique, nous avons adapté un algorithme en C trouvé sur Internet, qui analyse ligne par ligne l'objet, permettant ainsi un gain considérable en efficacité.

4 Bugs - Améliorations

Il subsiste de nombreux bugs et améliorations possibles de notre projet. En particulier, de nombreuses améliorations que nous pensions apporter à notre projet n'ont pas été implémentées par faute de temps, certains problèmes ayant été plus chronophages que prévu.

4.1 Améliorations

- Différents fluides (air, eau, vide ...)
- Variation aléatoire et incidence du vent
- Amélioration du modèle physique (élasticité ...)
- Modification du décor et de l'objet en temps réel par l'utilisateur
- Tracé de la trajectoire de l'objet

4.2 Bugs

- Lenteur de l'exécution sous Linux (pas de problèmes sous Windows)
- Le décor ne se remet pas à jour si on le redessine
- Calcul des tangentes peu précis (sinon il serait beaucoup trop lent)

5 Conclusion

Miam !

6 Annexes

6.1 Avis personnels et problèmes rencontrés

Tristan :

- Communication (qui fait quoi, quand et pourquoi...)
- Conflits SVN
- Implémentation pascal/Lazarus avec Linux = caca (pourquoi on fait pas du Cééééééééééééé...)

Rémi :

- Les autres ont rien foutus

Victor :

- Idem

Théo :

- niah

6.2 Diagrammes de classes

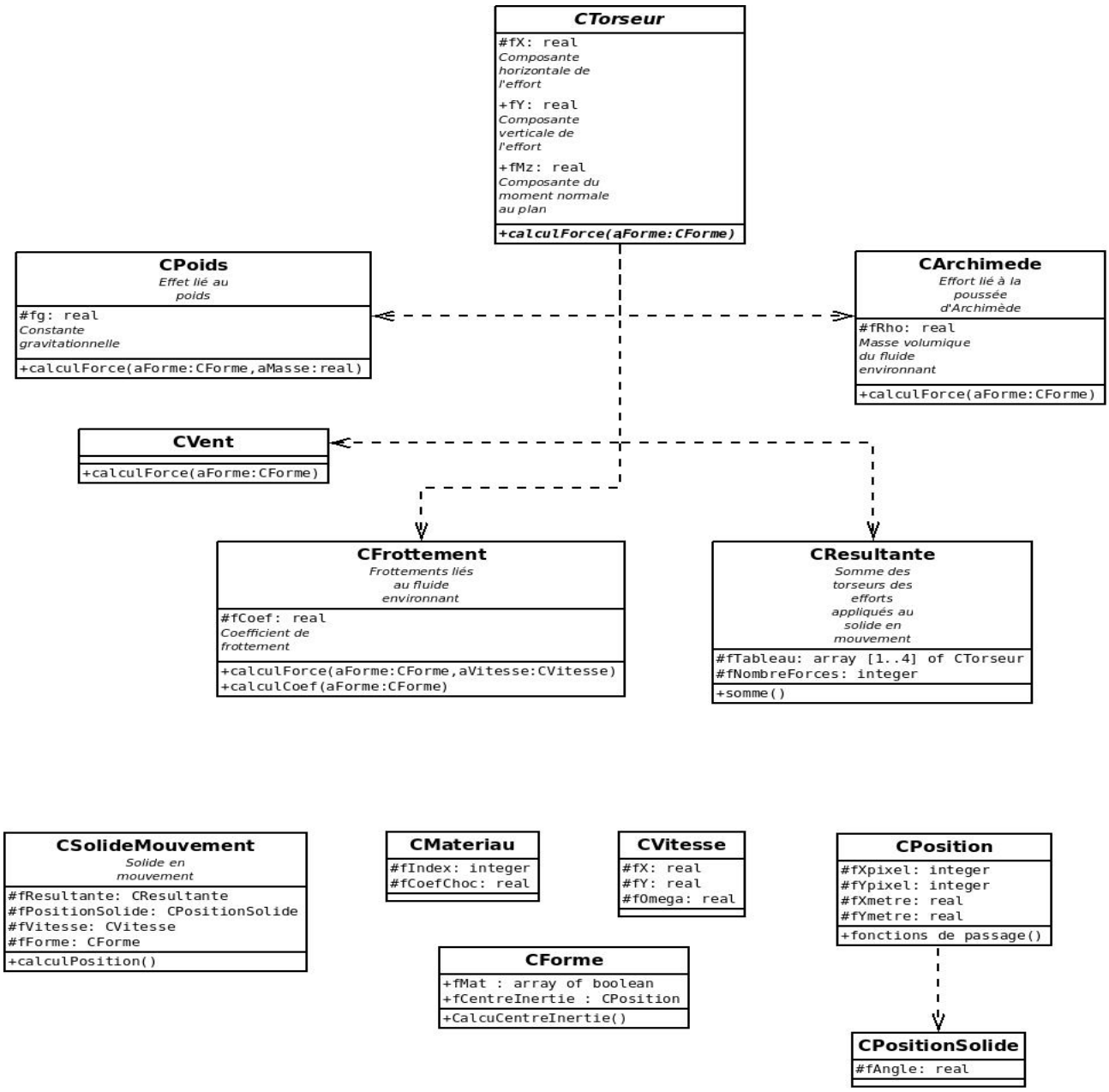

FIGURE 1 – Diagramme de classe utilisé pour notre projet

6.3 Impressions écrans

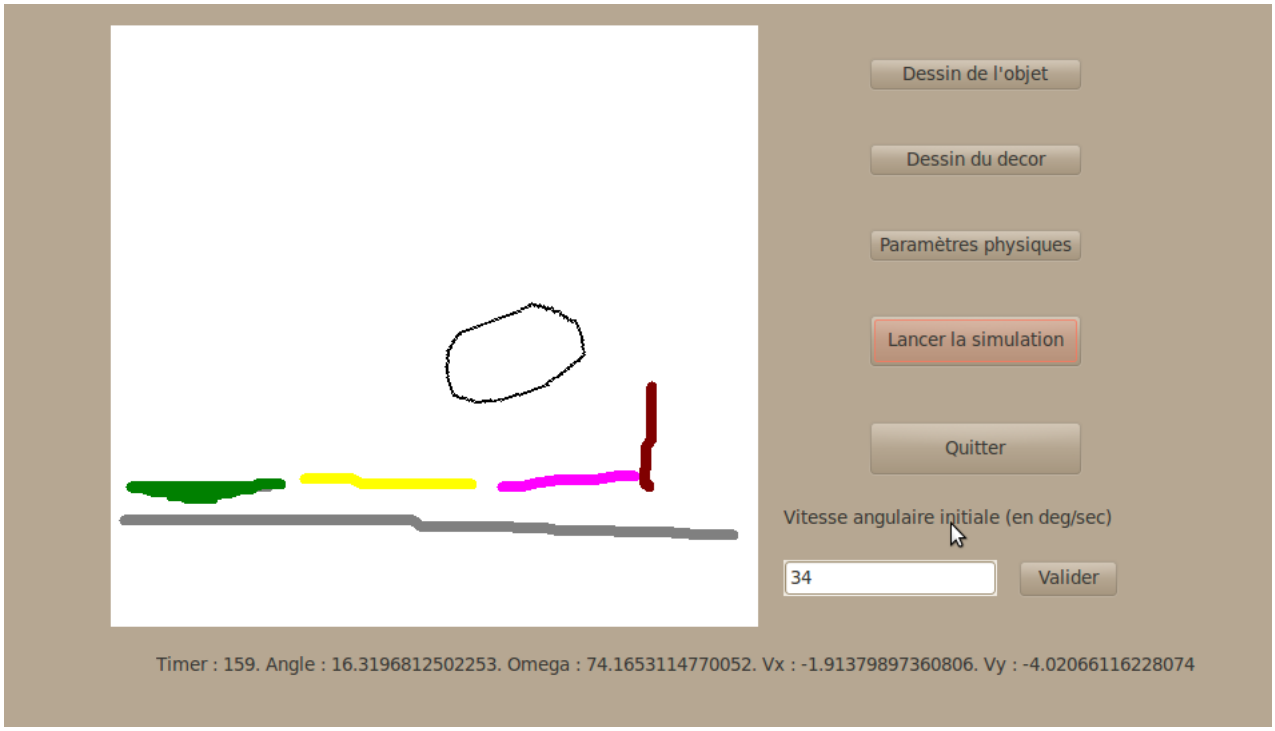

FIGURE 2 – Fenêtre principale

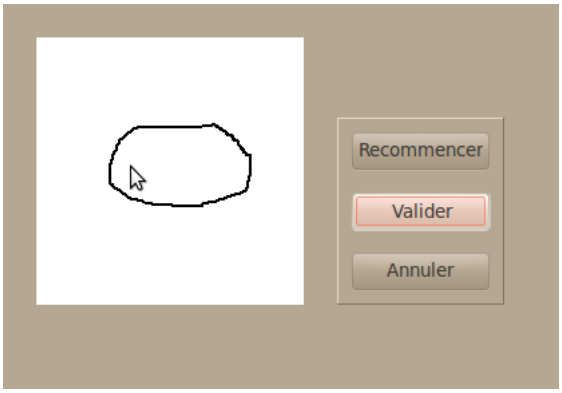

FIGURE 3 – Fenêtre gérant le dessin de l'objet

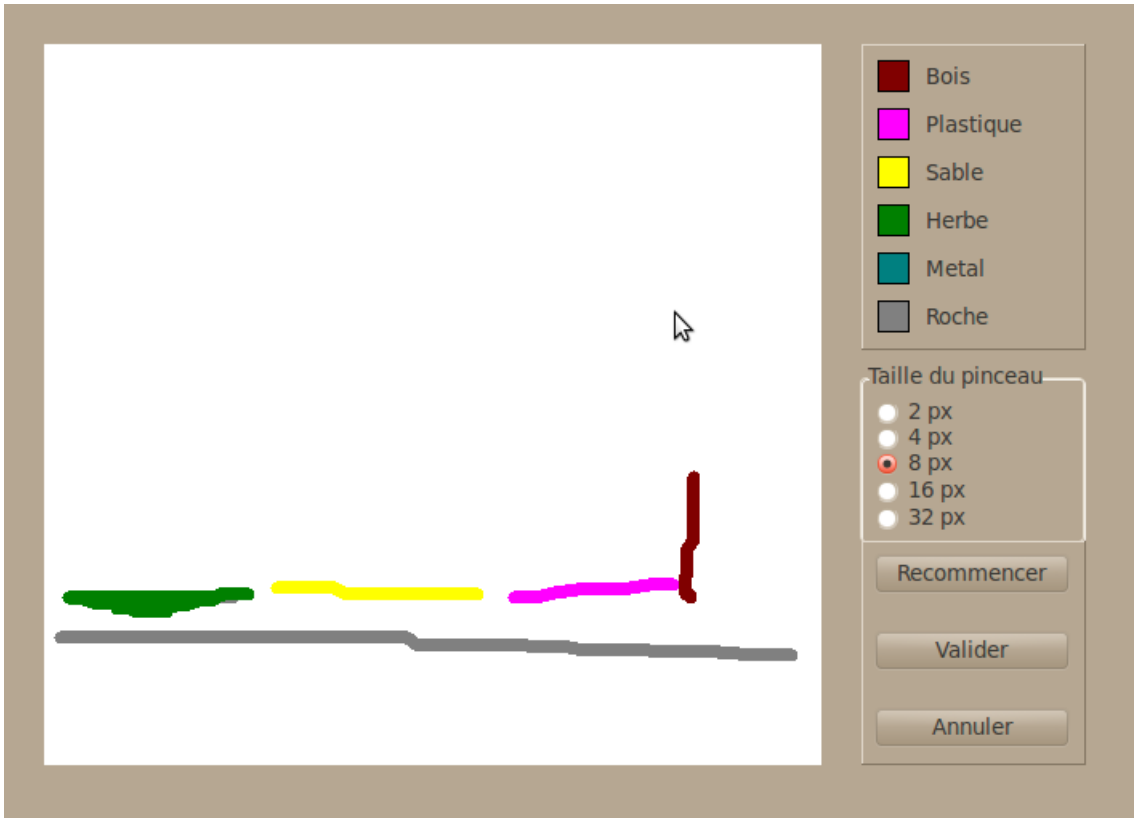

FIGURE 4 – Fenêtre gérant le dessin du décor