Université de Reims

Projet de programmation multimédia

Jeu de course futuriste

Paul Demeulenaere
04/05/2009

I. Circuit

Les modèles de circuit sont basés sur des B-Spline uniformes fermées d'ordre N. Elles permettent, à partir d'un petit nombre de points, de générer des courbes plus ou moins lisses (en fonction du degré). Ici, on dispose de deux polygones, l'un représentant le bord droit du circuit, l'autre le bord gauche. Ci-dessous, un exemple simple de fichier de circuit contenant uniquement quatre couples de points.

```
#Degré de la spline
3

#Nombre de couple de points
4

#Liste de points
-1000 0 0
-980 0 0
0 1000 0
0 980 0
1000 0 0
980 0 0
0 -1000 0
0 -980 0
```

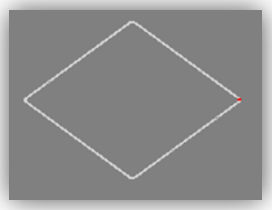


Figure 2: Degré 1

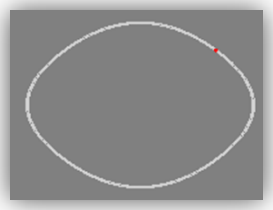


Figure 3 : Degré 2

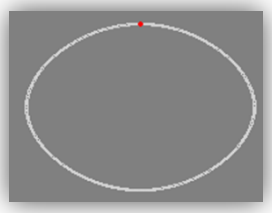


Figure 1 : Degré 3

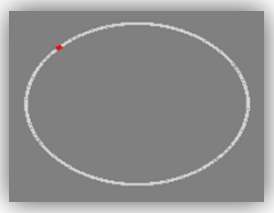


Figure 4 : Degré 4

Les polygones du circuit sont eux-mêmes échantillonnés en fonction de ces deux B-Spline. Il est possible de changer facilement cette valeur ; ci-dessous, plusieurs rendus de la même portion de circuit à des échantillonnages différents.

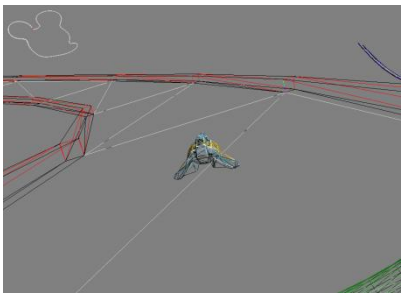


Figure 7 : 4 subdivisions

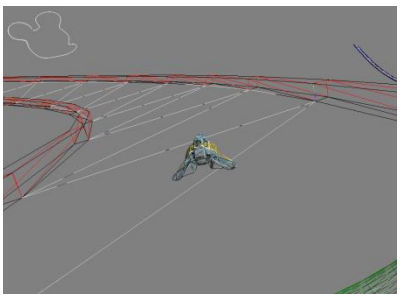


Figure 6 : 8 subdivisions

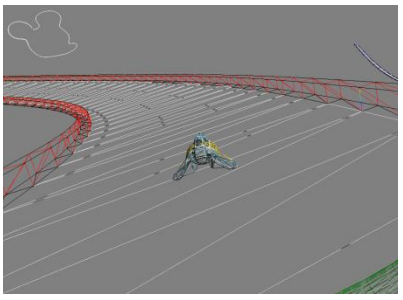


Figure 5 : 16 subdivisions

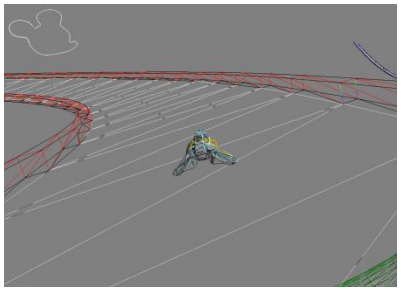


Figure 9 : 32 subdivisions

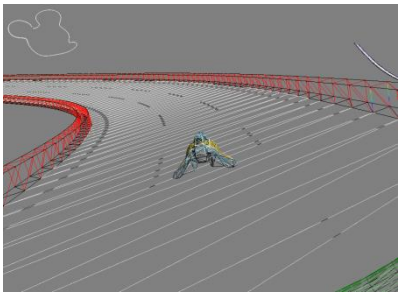


Figure 8 : 64 subdivisions

II. Vaisseau

Les vaisseaux sont chargés à partir de fichiers Wavefront OBJ. Plusieurs effets sont ensuite appliqués à l'affichage.

Figure 10 : Aperçu de l'ensemble des vaisseaux (rendu réalisé sous 3dsmax)

Tout d'abord, ces objets sont texturés. Dans l'application, pour simplifier la configuration, le nom du fichier obj est identique au nom du fichier texture (au format png).

Figure 11 : Exemple de texture

Une trainée de fumée a été rendue derrière le vaisseau. Elle suit son déplacement en mettant à jour un ruban à chaque fois que l'animation est jouée. La largeur du ruban dépend aussi de la vitesse du vaisseau.

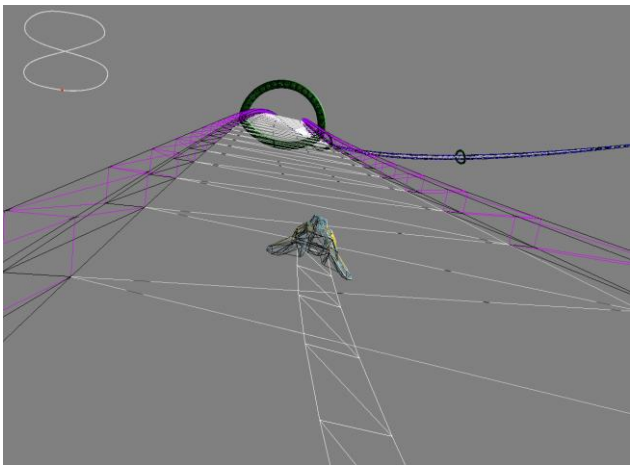


Figure 13 : En mode filaire

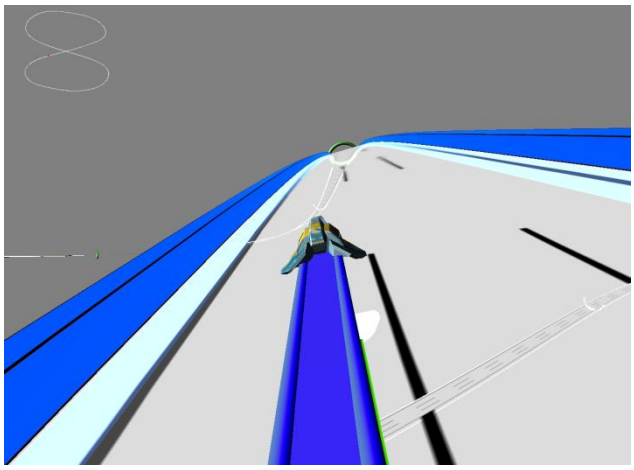


Figure 12 : normale calculée

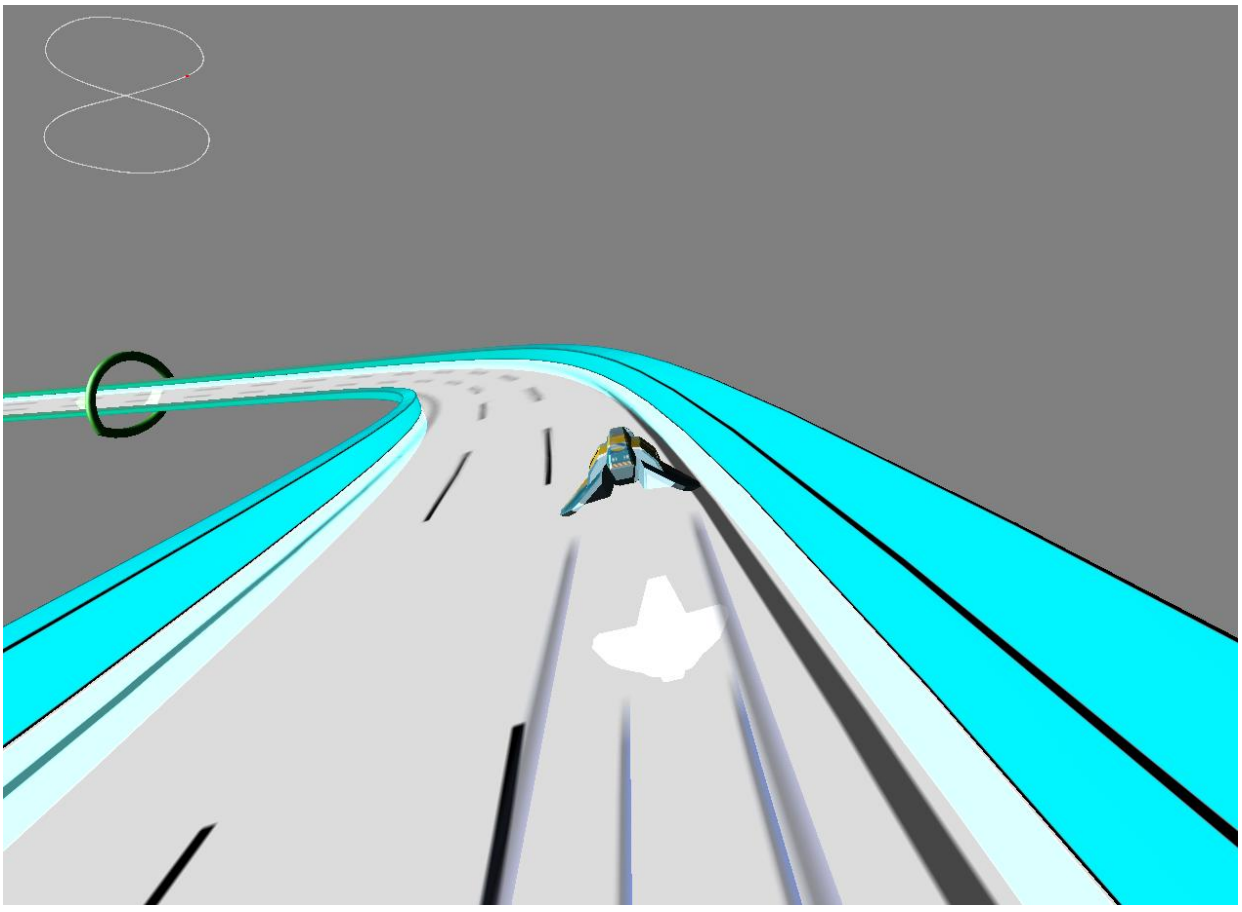


Figure 14 : Résultat après éclairage et transparence

Quand le vaisseau touche l'un des bords ou le sol, un effet de glow lui est appliqué, selon la même technique appelée « post GlowBalloon » disponible sur le site de NVidia. Cette technique consiste à dessiner une première fois l'objet normalement puis en l'agrandissant un peu avec un pixel shader particulier dans le but de créer une enveloppe transparente.

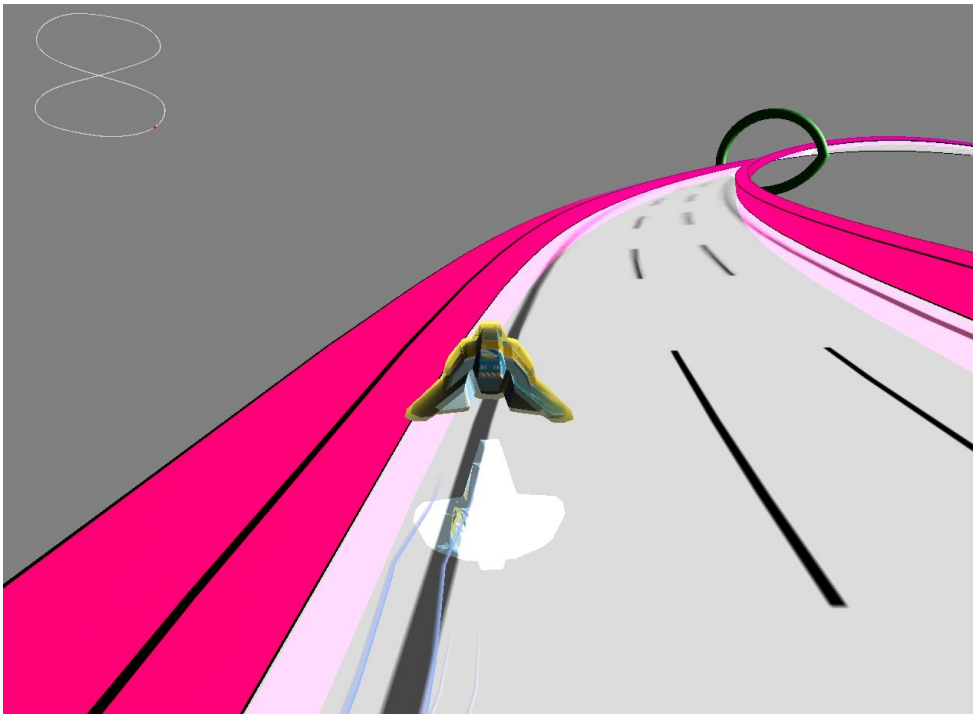


Figure 15 : Glow sur un vaisseau

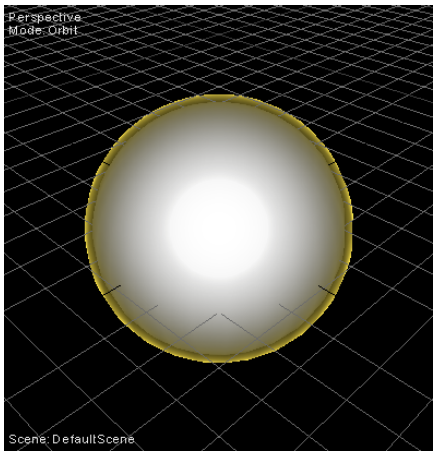


Figure 16 : Glow sur une sphère (depuis Fx composer)

III. Physique

La physique a été extrêmement simplifiée dans ce projet. Il n'y a même pas de vecteur de gravité. En réalité, un ressort est simulé entre la piste et le vaisseau. Il n'est pas nécessaire de tester les collisions avec tous les éléments de la piste, c'est pourquoi le circuit est découpé en portions.

Chacune des portions contient un centre et trois plans, un plan pour le sol, et deux pour les bords. Quand on cherche à appliquer la physique sur un vaisseau, on cherche la portion la plus proche de ce dernier pour réaliser la collision entre trois plans et une série de points correspondant aux bords du vaisseau.

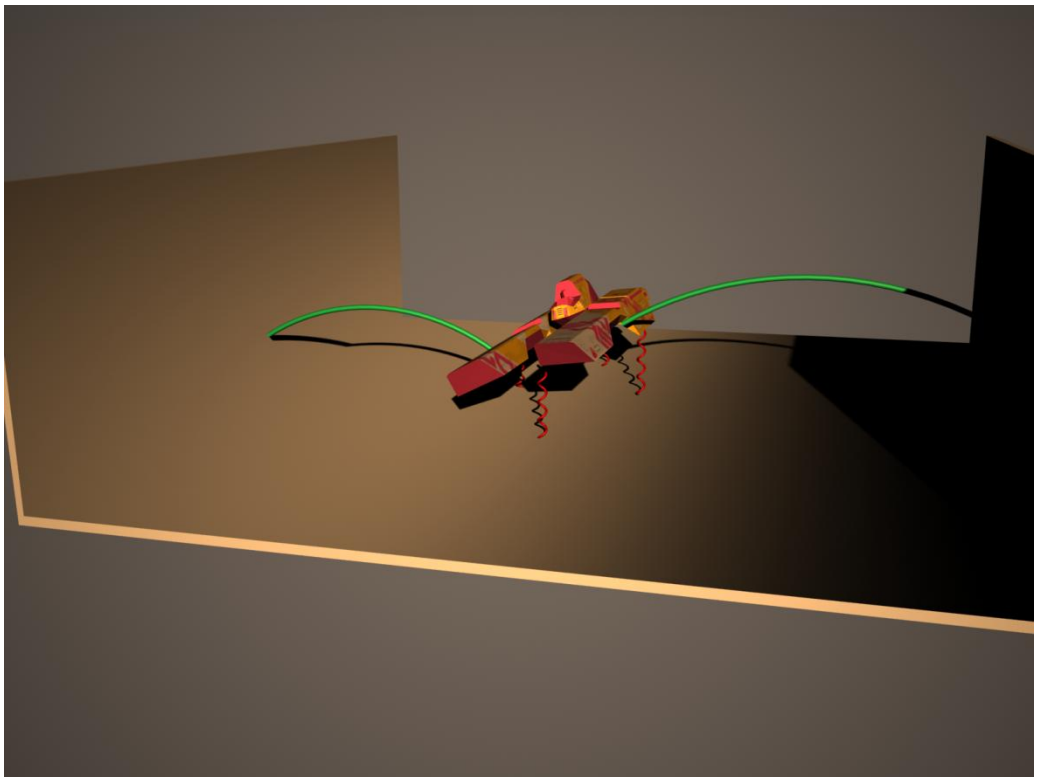
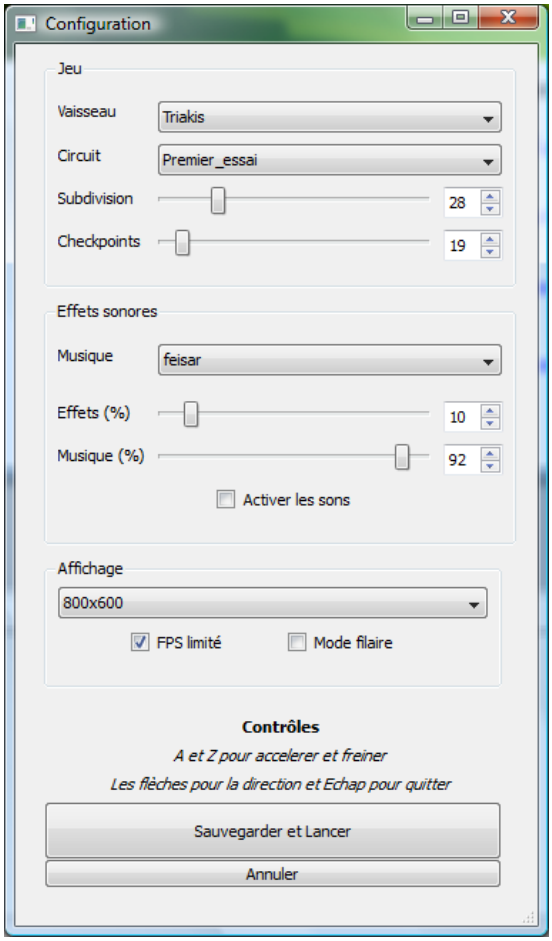


Figure 17 : Aperçu du fonctionnement des collisions avec la piste

IV. Configuration

Les circuits sont très simples à réaliser et très légers en mémoire, il aurait donc été dommage de ne pas pouvoir en changer facilement. C'est dans cette optique qu'une petite application QT a été créée ; elle s'occupe de charger et modifier les fichiers de configuration de l'application. Par son intermédiaire, il est possible de choisir un circuit, un vaisseau, régler les volumes sonores (voire les désactiver pour des soucis de performance) et de configurer l'affichage. Après la sauvegarde, l'exécutable est lancée automatiquement par l'intermédiaire d'un fichier de commande bat.



V. Bilan

Ci-dessous, un tableau récapitulatif mais pas forcement exhaustif des implémentations de l'application.

Objet	Réalisation
Contrôles	Oui, mais seulement clavier, pas de contrôleur…
Chargement de fichiers PLY	Oui (enfin, obj)
Textures	Oui
Sons 3D - OpenAL	Oui
Streaming - OpenAL	Oui
Anti-aliasing	Non
Shaders	Oui (HLSL et effets)
Eclairage par pixels	Oui (sur les tores et le vaisseau)

Par ailleurs, plusieurs points restent à développer. Tout d'abord, la recherche de la portion la plus proche lors du calcul de la physique se fait de manière séquentielle alors qu'il serait clairement plus rapide de réaliser cette recherche de manière dichotomique. Ensuite, un effet de réflexion a été implémenté en utilisant le stencil buffer et des objets dessinés à l'envers. Le problème est que la piste n'est pas dessinée dans le tampon de profondeur, ainsi, elle est visible quand elle se chevauche elle-même.

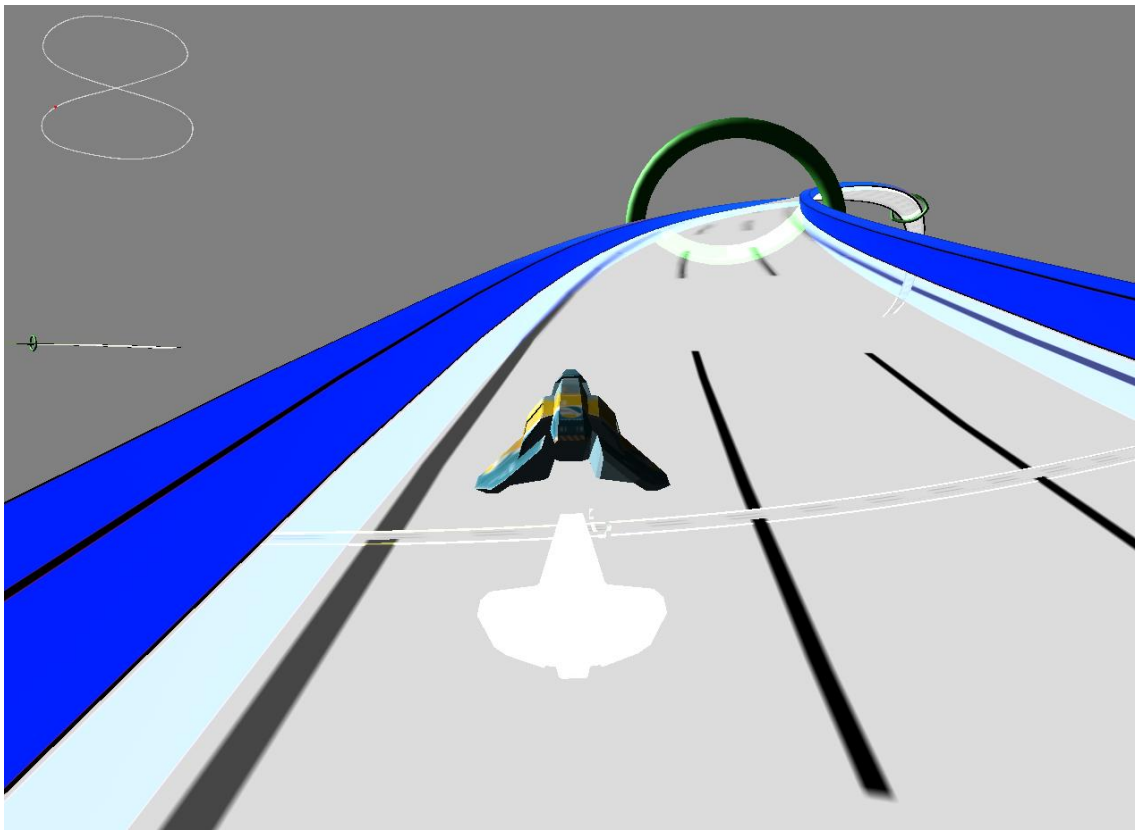


Figure 18 : Mise en évidence du problème de reflexion

En outre, l'application s'est révélée un peu plus rapide lorsque l'affichage n'était pas limité au rafraîchissement de l'écran (`D3DPRESENT_INTERVAL_IMMEDIATE` ou `D3DPRESENT_INTERVAL_ONE`) alors que les mises à jour des positions sont réalisées à intervalle régulier. Enfin, les desallocations ont parfois été oubliées et il resterait des éléments de gameplay à apporter comme le calcul du temps au tour ou des caméras differentes par exemple.