Projet - Modélisation Géométrique : Création et suivi de trajectoire de caméra

Alexis Gosselin, Issam Alouane

23 mai 2023

1 Introduction

L'objectif est de construire à partir de quelques points un module Unity3D qui permet de réaliser un suivi de trajectoire de caméra. Une trajectoire se décrit par un ensemble de points. Chacun de ces points représente une position et une rotation. Ainsi, pour décrire la trajectoire de la caméra, on utilisera les quaternions qui sont particulièrement adapté pour représenter l'orientation et la rotation dans l'espace tridimensionnel. Le suivi de trajectoire se décompose en 2 partie :

— le suivi de position : pour cela nous avons utilisé l'algorithme de subdivision de courbe spline uniforme implémenté au TP3 ou l'algorithme de DeCasteljau implémenté au TP2.

— le suivi de rotation : pour cela on doit réaliser une interpolation sphérique.

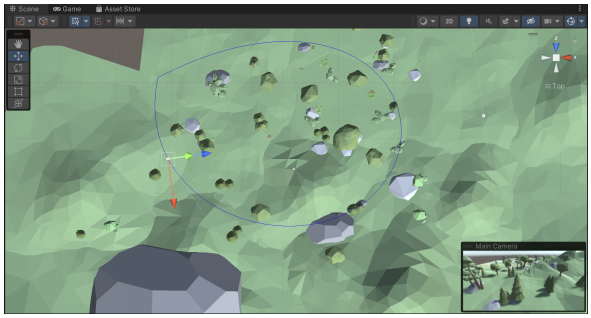

Figure 1: Trajectoire de la caméra avec DeCasteljau

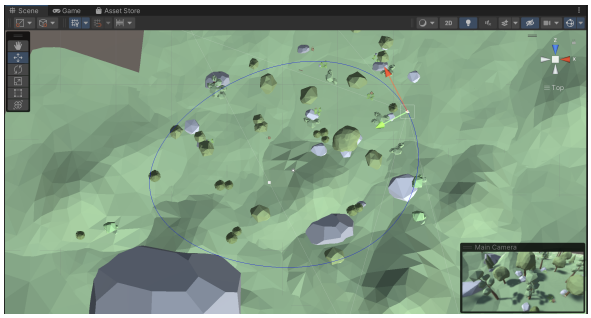

Figure 2: Trajectoire de la caméra avec courbe spline

2 Algorithmes utilisés

2.1 Suivi de position

Pour le suivi de position, on laissera le choix à l'utilisateur entre les 2 algorithmes proposés pour l'interpolation du chemin qui va être traversé par la caméra. Le point commun entre les deux approches est la tentative d'échantillonner la trajectoire avec un pas très petit afin d'obtenir un grand nombre de points. Cela permet de rendre le chemin plus fluide et continu, évitant ainsi les sauts ou les saccades perceptibles.
Pour l'algorithme de DeCasteljau, nous avons pris un pas d'échantillonnage de 1/10000f, alors que pour l'algorithme de courbe spline fermé nous avons pris un nombre d'itérations égale à 2000 avec un degré égal à 5.

2.2 Suivi de rotation

Pour le suivi de rotation, les quaternions ont été utilisés. À chaque point parcouru le long du chemin, la direction vers le centre estimé du chemin est calculée, puis la caméra est orientée de manière à pointer vers ce centre. Nous avons aussi utilisé les fonctions LookRotation et Slerp. La fonction LookRotation permet de calculer la rotation nécessaire pour faire face à une direction donnée, tandis que la fonction Slerp permet d'interpoler en douceur entre deux rotations afin d'assurer des transitions fluides de l'orientation de la caméra.

3 Avantages, inconvénients et limitations de ce modèle

Avantages :

— La trajectoire de la caméra est définie en seulement quelques points.

Inconvénients et limitations de ce modèle :

— Limité à des trajectoires de caméra en rotation, la caméra ne peut pas aller tout droit par exemple (sauf avec DeCasteljau mais la caméra est toujours dirigée vers le centre).

— Nécessite un grand nombre de points pour observer une trajectoire fluide, on peut donc avoir des problèmes de stockage et de complexité.

— Trajectoire de la caméra uniquement dans un plan 2D.

4 Conclusion

Le script de suivi de trajectoire de la caméra développé présente des perspectives d'amélioration intéressantes. Tout d'abord, il serait possible d'étendre la trajectoire de la caméra en 3D au lieu de se limiter à un plan. Cela permettrait d'explorer des environnements plus immersifs et d'ajouter une dimension supplémentaire à l'expérience. De plus, il serait bénéfique d'ajouter des fonctionnalités supplémentaires telles que la possibilité de faire pivoter la caméra vers le haut ou vers le bas à l'aide d'un bouton ou de choisir facilement l'altitude souhaitée.

En ce qui concerne l'algorithme de DeCasteljau, des améliorations pourraient être apportées pour obtenir une trajectoire plus circulaire vers la fin. Cela contribuerait à une transition plus fluide et esthétique entre les points de contrôle.

Lien vers le projet complet