Rapport animation Nicolas Pronost

Barros Mikel-ange

I) Présentation du TP

L’objectif de ce tp est d’animer un bipède de manière réaliste en fonction des moments des articulations du bipède ainsi que des forces extérieures du modèle. Afin de mettre en place cela, nous définissons notre personnage comme étant un corps rigide articulé composé d’un ensemble de membres reliés entre eux par des articulations. Chacun des membres du personnage aura son propre poids et ses propres dimensions. Ainsi à partir de ces différentes informations et des forces appliquées sur notre personnage, nous pouvons définir des moments cinétiques pour chacune des articulations ce qui lui permettra de rester au sol. Le moment cinétique est une force appliquée dans le cadre d’une rotation et qui va avoir les mêmes propriétés que la quantité de mouvement dans le cadre d’une translation. Il va permettre d’appliquer une résistance aux différentes forces perçues par le personnage et donc lui permettre de rester debout. La formule du moment cinétique en un point M par rapport à un point O est décrite comme étant le produit vectoriel entre la quantité de mouvement du point et le vecteur OM. Afin de vérifier la validité de l’animation, nous testerons la distance de déplacement du bipède ainsi que sa stabilité, et ce, malgré des événements visant à le faire tomber tels que l’ajout de marche sur son passage ou l’envoie de balle sur son corps, qui naturellement appliquera de nouvelles forces au modèle.

II) Introduction

Que ce soit pour représenter des événements climatiques ou juste le comportement d’un objet ou d’un animal , la physique a toujours été un point essentiel de l’animation. Et il en va de même pour l’animation de personnages réalistes et notamment pour ce qui concerne le contrôle de mouvement. En effet, dans les jeux vidéo, le personnage est animé par deux grands contrôleurs : le premier va lui faire prendre des positions prédéfinies telles que la marche ou le saut, alors que le deuxième va le faire réagir en fonction des différentes actions qui se passent autour de lui. Ces deux contrôleurs agissent ensemble afin de donner aux jeux vidéo un rendu le plus réaliste possible.Dans ce document, nous nous pencherons sur les deux contrôleurs et leur complémentarité. Comme dit dans la présentation du TP, nous allons chercher à modéliser des positions réalistes en fonctions des forces appliquées sur le personnage. Afin de tester le contrôleur et de l’améliorer, nous pouvions créer de nouveaux mouvements,améliorer le contrôle du personnage ou toute autre idée permettant de rendre le résultat plus intéressant et utilisable . Je vais détailler les différentes améliorations choisies et leur fonctionnement dans les deux parties suivantes.

III) Choix des améliorations

III.1) Ajouts de différentes machines à état de mouvements

En tout, trois machines à état ont été rajoutées, une pour le saut, une pour donner un coup de pied et une pour marcher à reculons. Ces trois machines à état ont été implémentées conjointement dans le but de pouvoir rendre un rendu réaliste. Afin d'obtenir les meilleurs résultats possibles, j'ai déterminé dans chaque cas les meilleurs paramètres pour la position en me basant sur le contrôleur par optimisation de dynamique contrainte fourni dans ce TP. Un contrôleur par optimisation de dynamique contrainte est comme son nom l'indique un moyen de contrôler les actions du personnage en cherchant à optimiser les paramètres pour optimiser une fonction de coût. Ce type de modèle est très pratique dans le cadre de la simulation basée physique, car il est souvent très fastidieux d'obtenir des paramètres optimaux pour notre simulation. La méthode mise en place ici est une méthode dite offline, c'est-à-dire que les paramètres sont optimisés sur une longue période puis appliqués au modèle par la suite.

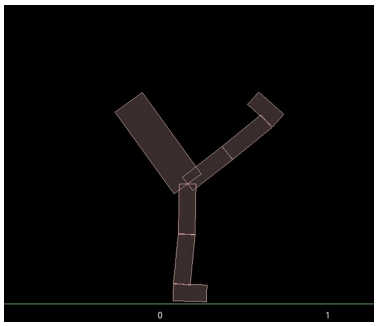

Figure 2: Coup de pied

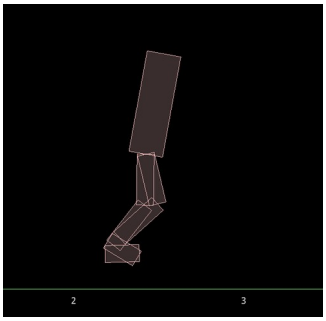

Figure 3: saut

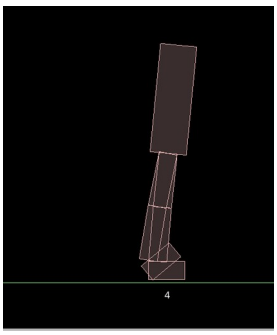

Figure 1: marche arrière

III.2) Ajout d'un contrôleur permettant de passer d'une machine à état à l'autre

Maintenant que nous avions plusieurs machines à état, il est devenu intéressant de développer un moyen rapide de passer d'une animation à l'autre et d'ainsi obtenir un rendu plus proche de ce que l'on pourrait obtenir dans de l'animation de personnages classique et ou dans un jeu vidéo.

IV) Mise en place des améliorations

IV.1) Ajouts de différentes machines à état de mouvements

Afin de créer une machine à état, j'ai dû redéfinir manuellement chacune des positions et chacun des angles de mes articulations pour chacune des positions. Comme expliqué plus haut, la difficulté ici a été de trouver les bons paramètres pour que chaque pose soit réaliste et n'impacte pas la stabilité de mon personnage.

IV.2) Ajout d’un contrôleur permettant de passer d’une machine à état à l’autre

Pour mettre en place cette méthode, nous avons revu la structure de donnée pour qu’elle contienne une liste de machines à état et que l’on change d’une machine à l’autre en appuyant sur un bouton. Cette approche simple, à permis d’obtenir une animation de personnages plus complexes et utilisable. Ça m’a aussi permis de tester la stabilité du modèle face à un changement brutal de position.

V) Difficultés rencontrées

V.1) Améliorations abandonnées

V.1.a) Déplacements dans l’eau

Afin de simuler les déplacements dans l’eau, deux idées me sont venu, une approche lagrangienne à base de sph et une approche eulérienne permettant de découper les zones d’eau en grille afin d’en calculer les forces d’interactions dans chaque zone aqueuse et d’appliquer des forces au personnage pour le ralentir ou en limiter les mouvements. Cependant, cette approche s’est avérée trop coûteuse et difficile à mettre en place pour une utilisation temps réel et a vite été abandonnée. J’avais aussi pensé à imiter un tel fonctionnement en atténuant l’effet de la gravité sous l’eau et en appliquant un courant contraire au sens de déplacement du personnage. Mais le résultat obtenu était assez peu réaliste et j’ai préféré abandonner cette piste bien qu’elle soit celle le plus souvent utilisée dans les jeux vidéo 2D.

V.2) Difficultés générales

Contrairement à l’année dernière, la plus grosse difficulté que j’ai rencontrée ne fut pas de comprendre le code et de le modifier, mais bien de trouver les bons paramètres et les bonnes fonctions de coût pour l’optimisation des dis paramètres.
Cependant, une autre difficulté que j’ai rencontrée, à été de comprendre chacune des améliorations proposées et certaines restent encore un peu floue.

VI) Conclusion

Ce Tp est une bonne approche de la physique appliquée à un personnage afin de lui faire avoir une attitude et des mouvements réalistes. De plus, il permet de découvrir et expérimenter sur un modèle plus ou moins simple comment sont appliqués et exploitées les différentes forces pouvant s’appliquer sur le corps. Cependant, il m’est vite apparu que pour obtenir des mouvements très réalistes et un fonctionnement cohérent du début à la fin de la simulation, de nombreux paramètres devaient être prédéfinis, connus et extrêmement précis ce qui rendait le travail en amont extrêmement compliqué et n’exclut pas la possibilité d’un bug ou d’un problème dans un monde aussi libre qu’un jeu vidéo par exemple. En effet, la complexité d’un modèle de contrôle de personnages évolue en fonction du nombre de forces qui s’appliquent dessus et pour un jeu vidéo cela peut très vite devenir compliqué à gérer. Et ce encore plus en considérant la quantité de calculs considérable à faire risquant de retarder l’exécution dans le cadre d’un projet en temps réel.