Projet de Mathématiques et Physique pour le Jeu Vidéo – Phase 3

Laurianna FERRA, Clément BOURLET, Mathis RONZON, Xavier PALIX, Léna HERAU
Groupe I

Déroulement de la Phase 3

La phase 3 a commencé avec un travail de refonte du système de boucles de la phase 2 afin de pouvoir séparer la boucle de mise à jour de la physique et la boucle de mise à jour du rendu, dans une volonté de garantir de bonnes performances tout en ayant une simulation réaliste. De plus, le système d'actors permettant de définir une liste d'entités basées sur des rigidbodies et une apparence physique offre une expérience réutilisable et modulable. En parallèle, la classe rigidbody a été implémentée grâce à l'expérience acquise durant l'élaboration de la classe particle. De même, une nouvelle interface de forces dédiée pour les rigidbodies a été développée en s'inspirant des précédentes interfaces de forces pour les particles. Par la suite, des éléments de démonstration ont été conçus dans le gameGUI en créant des actors suivant un rigidbody défini par une position, une orientation, une masse et un tenseur d'inertie. Leur rendu a été implémenté dans le gameGUI et l'implémentation d'impulsions nécessaires au démonstrations demandées ont été faites dans windowPart3 (c'est-à-dire des forces qui ne proviennent d'aucune force réaliste).

Résumé des implémentations

Parmi les nouvelles implémentations, nous avons en premier lieu celle de la refonte du système de boucles. Après la phase 2, un problème de performances des machines amenait à notre blob à exploser car le délai de traitement étant trop long, les forces appliquées sur les ressorts s'accumulaient jusqu'à briser les liens qui retenaient les particules du blob ensemble. Parti de ce constat, la windowPart3 a changé de formalisme par rapport à windowPart2. Les boucles de physique et de rendu sont traitées séparément grâce à frame rate différents sur chaque aspect. En effet, la boucle de physique est appelée un frame rate bien plus haut que celui de la boucle de rendu.

Ensuite, nous avons l'implémentation de la classe physicSystem qui permet de donner à windowPart3 l'entièreté du système physique en boîte noire. Dans cette classe, les forces sont appliquées à la liste d'actors du système physique sans que jamais windowPart3 n'est à connaissances de ces forces, quel soit classiques (comme la gravité, les forces de ressorts etc.) ou émanant des entrées claviers de l'utilisateur (la force d'un saut, la force de déplacement etc.). À l'exception bien sûr des impulsions nécessaires à la création des démonstrations qui, elles, sont codées en dur dans le windowPart3.

Par ailleurs, l'implémentation des actors permettent d'ajouter n'importe quel type d'objets possédant un rigidbody tout en gardant la main sur son aspect de rendu. Le système physique connaît et met à jour alors la liste des actors de la scène en passant par la classe générale rigidbody.

Ainsi, nous avons aussi implémenté la classe rigidbody. Cette classe assez généraliste prend uniquement en compte les considérations physiques d'un corps rigide. Elle contient donc la position du centre de masse dans le repère monde, l'orientation du repère local dans le repère monde au

format quaternion, la matrice inverse de son tenseur d'inertie, sa masse inverse ainsi que les vélocités/accélérations linéaires et angulaires. Elle permet aussi de réaliser un ajout de forces en considérant un point autre que celui du centre de masse de l'objet à la fois avec ce point défini dans le repère local ou dans le repère monde. Finalement, elle gère aussi la mise à jour de la matrice inverse du tenseur d'inertie dans le repère monde via un calcul de matrice de transformation homogène et intègre physiquement le rigidbody en considérant la physique rotationnelle.

De plus, nous avons implémenté une nouvelle interface de forces pour les rigidbodies en s'appuyant sur celle conçue pour les particles, tout en considérant à présent la possibilité d'appliquer une force sur un point autre que celui du centre de masse. Ainsi, nous avons implémenté de cette interface trois générateurs de force : celui de la gravité, de la force des ressorts et des forces correspondantes aux entrées clavier de l'utilisateur.

Enfin, pour afficher les actors de la scène, nous avons programmé des méthodes dans le gameGUI permettant de définir une liste de Polygones. Cette liste est ensuite passée à l'initialisation de l'actor avec le rigidbody correspondant et lors de la boucle PaintGL, le gameGUI parcourt la liste des actors pour appeler la fonction drawPolygone à partir de sa liste de Polygones.

Particularités rencontrées

Aucune particularité autre que certaines difficultés qui ont été résolues sans apporter de changements majeurs à notre projet n'a été rencontrée lors de cette phase.