DM programmation interface Ocaml :Cut The Rope

Enjeu 1:

Trouver une modélisation pour simuler la physique d’une balle attachée à une ou plusieurs cordes.

*Solution 1:*

Découpage d’une corde en plusieurs points reliés par des ressorts.

| Avantages | Inconvénients |
| --- | --- |
| * Très réaliste * Gestion optimale des caractéristiques de la corde * Permet les effets de rebond | * Très coûteux en ressources * Implémentation de frottements difficile * Corde unique |

*Solution 2:*

Disjonction de cas entre corde tendu et corde pendante:

* toujours qu’une unique corde tendue (la dernière qui s’est tendue)
* les autres cordes, dîtes pendantes, sont simulées par une fonction de la forme: + c

| Avantages | Inconvénients |
| --- | --- |
| * Plutôt réaliste * Assez peu coûteux * Permet d’attacher plusieurs cordes avec frottements * Solution numérique de f existante (<https://math.stackexchange.com/questions/3557767/how-to-construct-a-catenary-of-a-specified-length-through-two-specified-points>) | * Pas de rebonds * Moins réaliste |

Enjeu 2:

Afficher un graphique d’une fonction de manière efficace.

Enjeu 3:

Simuler la gravité sur les points mobiles (principalement la balle).

*solution 1:*

Ajouter à la vitesse l' accélération fois le temps passé depuis la dernière mise à jour: approximation d’une série de Taylor à l’ordre 1, ou aussi appelé approximation par la tangente.

*Solution finale aux 3 enjeux précédents*

Utilisation de "Varlet's integration" pour simuler la physique. Cette méthode combine un nombre limité de calcul pour un résultat qualitatif, tout en restant raisonnablement compliqué à intégrer en Ocaml.

Enjeu 4:

Importer des images, voir des gif (sous format "png sequence") de manière efficace avec un affichage peu coûteux sur Ocaml.

*solution :*

L'import direct des png étant trop compliqué, un format ".brc" (ficher txt renommée pour les reconnaître) à été mis au point, ce dernier est créé grâce à python. Grâce aux modules PIL et numpy, le png est transformé en tableaux de 4-uplet contenant les valeurs :

* R (quantité de rouge) de 0 à 255
* G (quantité de vert) de 0 à 255
* B (quantité de bleu) de 0 à 255
* A (transparence) de 0 à 255

Ce format est lu en Ocaml transformé en liste de couleur graphics, puis interprèté comme une image par la fonction make\_image (on notera que cette dernière nécessite la fenêtre graphique ouverte et que l'image crée à un fort taux de corruption lors de la fermeture de la fenêtre, ainsi le processus d'import à été automatisé, et l'import est la première étape au lancement du jeu).