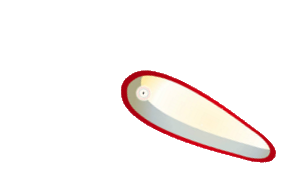
Pinball GUYARD Antonin

L’objectif de notre programme à Léo et à moi-même est de créer un jeu de flipper sur ordinateur en utilisant Python. Pour cela nous avons utilisé de nombreux Logiciels tels que :

-Python (et son extension Pygame)

-Paint.net (pour le traitement des images

Composition de l’interface :

* 3 bumpers
* 2 Flippers

Q:\Espace d'échange\Projets ISN\pinball\voltball.png

* Une balle

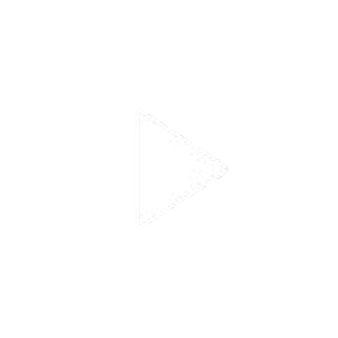
* Un cadre d’affichage du score

Nous avons choisi des images originales pour composer notre flipper car nous voulions créer une interface différente et amusante (ce qui est le but premier d’un jeu).

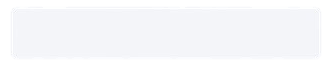
Nos attentes et objectifs :

Nous souhaitons pouvoir faire fonctionner les deux flippers, pouvoir lancer une balle qui rebondirai sur les différents éléments du jeu, ainsi que comptabiliser le score obtenu par le joueur.

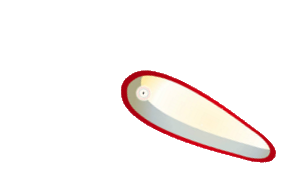
De plus, nous avons ajouté une musique de fond afin de rendre l’expérience encore plus intéressante.

Fonctionnement basique du programme :

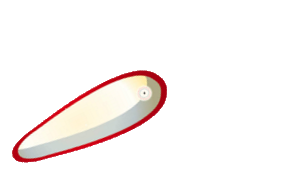
Au lancement du programme, une musique se déclenche (crazy frog)

Q:\Espace d'échange\Projets ISN\pinball\voltball.png

Quand on appuie sur la barre d’espace, on lance la balle qui était auparavant immobile



Quand on appuie sur la flèche gauche du clavier, le flipper gauche se met en rotation jusqu’à obtenir un angle de 70° par rapport à sa position initiale.





Quand on appuie sur la flèche droite du clavier, le flipper droit se met en rotation jusqu’à obtenir un angle de -70° par rapport à sa position initiale.

Répartition des taches :

Léo s’occupera de l’affichage du score et de sa zone particulière, de l’insertion de la musique

Antonin s’occupera de la mise en place du flipper et de l’articulation des différents éléments.

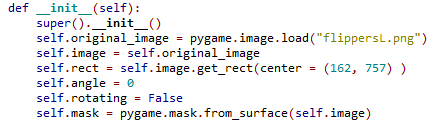
Le mouvement de la balle (en raison de sa complexité) est géré par les 2 élèves.

Détail de certaines fonctions complexes :

Nous avons créé des sprites pour les flippers et la balle, par exemple, pour le flipper, nous avons :



La fonction Initiale :



On définit (dans l’ordre) :

- l’image d’origine

- l’image à utiliser

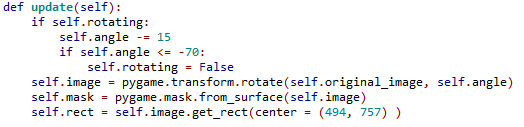
- le rectangle et son centre (utile pour les rotations)

- l’angle initial

- la situation initiale de la fonction rotating

- le mask afin de ne pas prendre en compte les zones vides du rectangle de l’image

La fonction update :



On y inscrit :

- La fonction rotating :

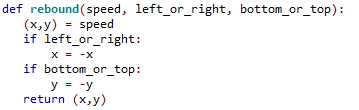
- on pivote le flipper de -15° à chaque frame

- si l’angle du flipper dépasse -70°, on définit la fonction comme fausse, elle n’a donc plus lieu et le flipper reste à l’Etat actuel

-on redéfinit l’image, le mask, et le rectangle afin d’actualiser le flipper dans l’interface

A présent, nous allons observer comment notre balle peut rebondir sur les bords de l’écran :

La fonction rebound :



On indique que lorsque la balle arrive sur l’un des bords de l’écran, on inverse la composante de la vitesse (x ou y, cela dépend de si la balle touche le bord droit/gauche, ou le bord haut/bas)

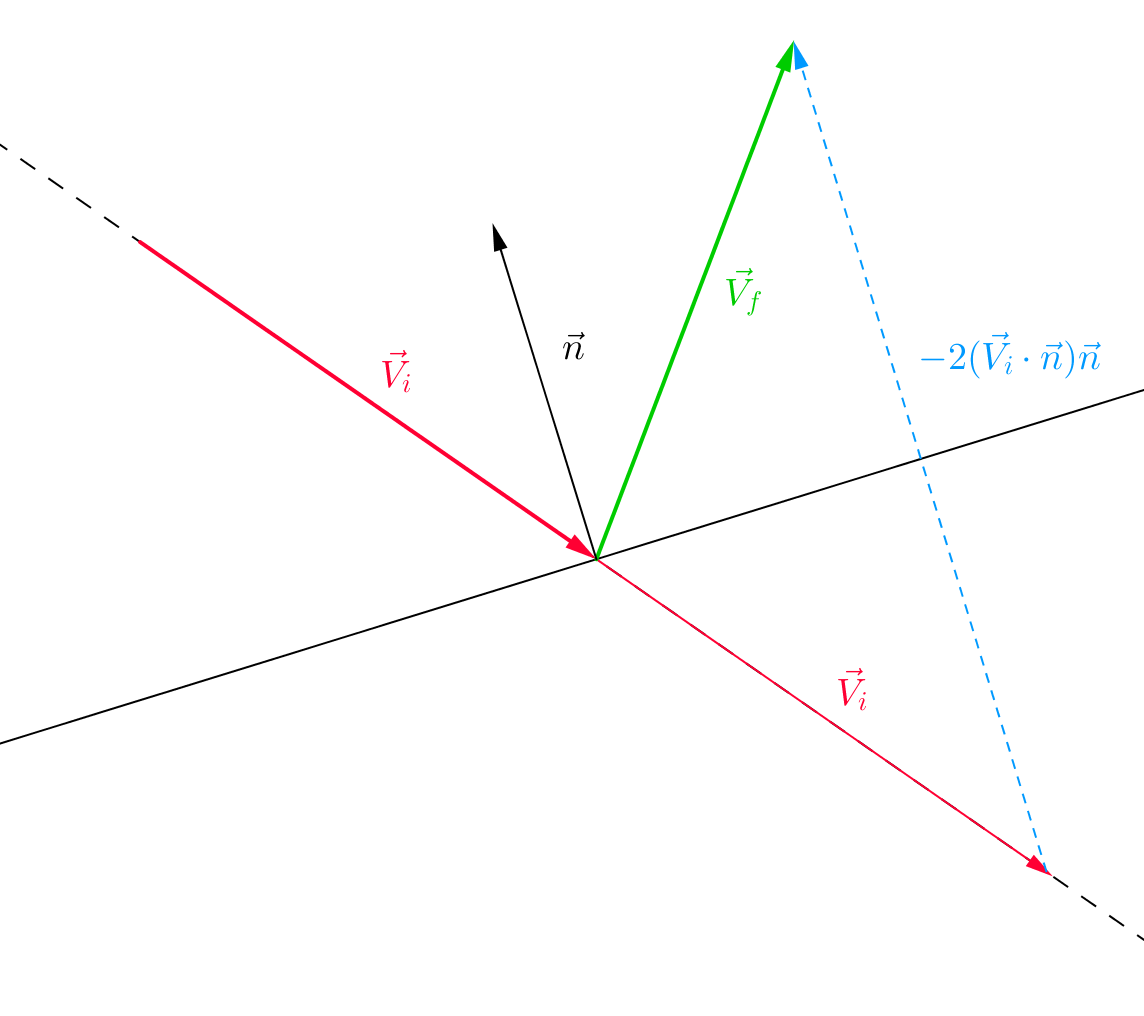
Difficultés rencontrées :

La plus grosse difficulté qui nous a stoppé dans notre progression est l’insertion des collisions plus complexes que celles de sur les bords de l’écran.

En effet, sachant que les flippers ne sont pas à une position horizontale ou verticale, nous ne pouvons pas appliquer le fonction rebound sur ces derniers.

Nous devons donc utiliser des produits scalaires et des calculs vectoriels afin d’obtenir des collisions plus précises.

Cependant, cette partie reste en cours de travail et n’est pas encore aboutie.



Impressions personnelles :

Je suis assez content d’avoir pu réaliser ce flipper.

En effet, cela m’a permis de comprendre un peu mieux comment fonctionne l’un de mes jeux préférés.

Mes impressions restent néanmoins nuancées puisque je n’ai pas encore pu insérer des collisions précises, ce qui en soit est une légère déception, puisque nous ne pouvons pas dire que notre flipper est terminé.