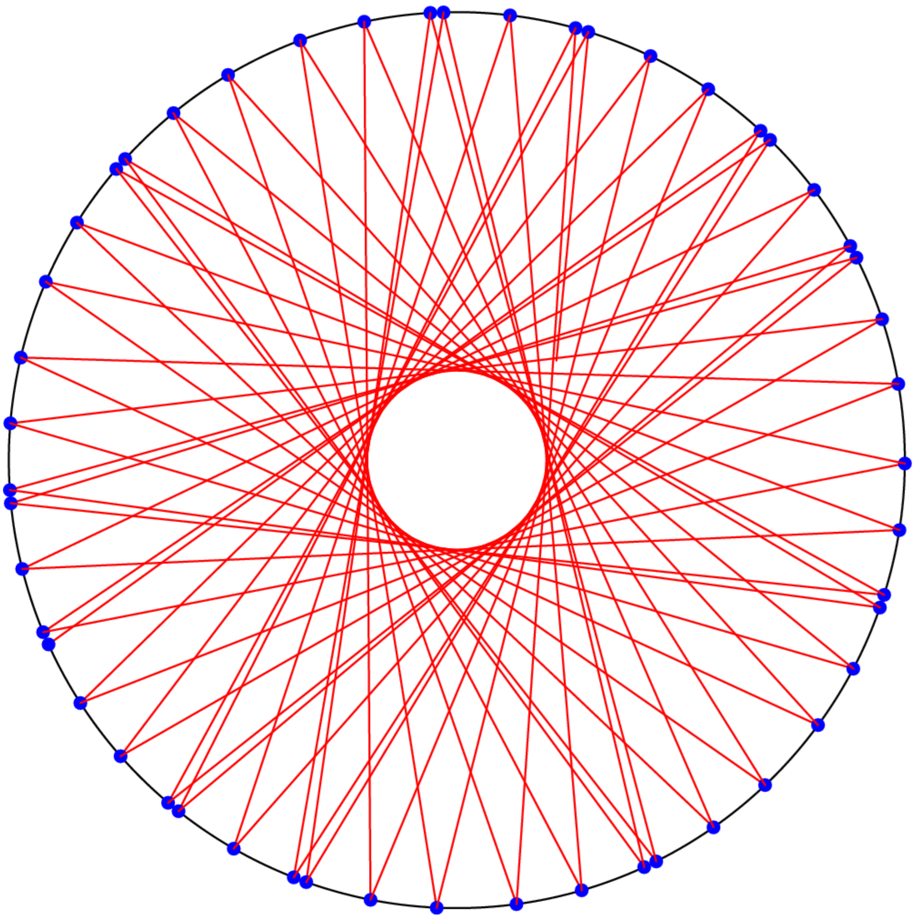
RODRIGUEZ Clément
BUDISAVLJEVIC Rémi

Méthodes numériques :Projet

Billard



Introduction

Le principe de ce projet consiste à étudier et modéliser l’évolution de la trajectoire d’une particule dans un billard circulaire, celle-ci subissant des réflexions successives sur les bords du billard. D’un point de vue plus général, cette étude illustre la sensibilité d’un système aux conditions initiales imposées.
Les buts sont donc de modéliser ces trajectoires, les étudier afin d’en déduire les effets des paramètres initiaux sur le système, et trouver les paramètres qui donnent certaines trajectoires particulières.

I/ Le problème physique

L’étude de cette problématique nécessite de modéliser le modèle physique d’un billard circulaire et d’une particule se déplaçant à l’intérieur. L’important est donc de trouver l’équation de la trajectoire de la particule, quelque soit sa position, et déterminer son comportement près un rebond.

On a donc pour cela considéré les coordonnées polaires, étant donné que nous travaillons avec un billard circulaire, cela semble plus approprié. On a par ailleurs assimilé la boule à un point, parce que ce qui nus intéresse vraiment dans ce projet est l’étude des trajectoires. Nous nous sommes donc intéressés à l’équation d’une droite en coordonnées polaires. Cette équation est la suivante :

$$r(\theta) = r_0 \frac{1}{\cos(\theta - \varphi)}$$

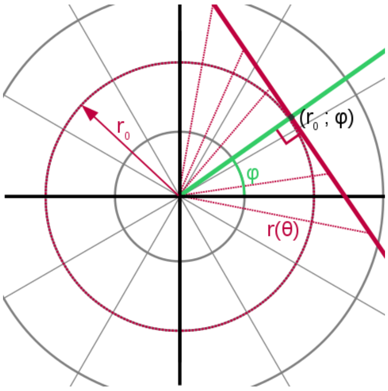
où r est la distance de la boule à l’origine, pour un certain θ.

θ est l’angle de la boule avec l’axe polaire.

r_0 et φ sont les coordonnées du point d’intersection entre la trajectoire, et sa perpendiculaire passant par l’origine.

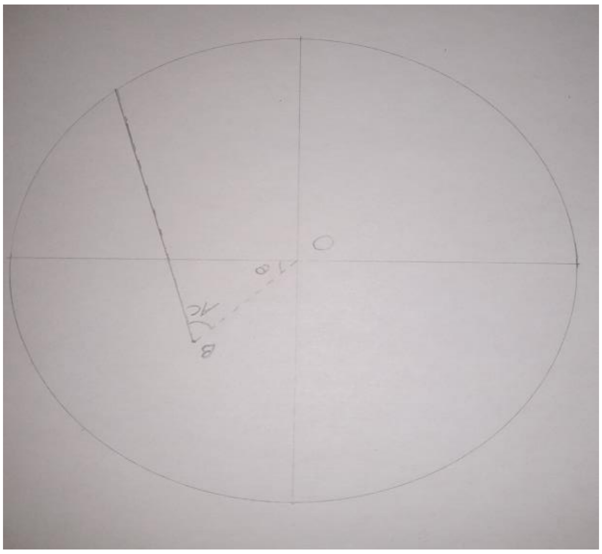
Or, dans notre cas, on veut avoir r = 1 (= le rayon du billard).
On cherche donc un θ, ce qui se traduit par l’équation suivante :

$\theta = \arccos(\frac{ro}{r}) + \varphi$



Le principe est donc de trouver cette équation au départ de la boule, puis pour chaque rebond, c'est à dire trouver le point de coordonnées couple (r_0, φ), qu'on appellera le point D.

En premier lieu, il faut définir « l'angle de coup », c'est à dire l'angle avec lequel la boule va être lancée au départ. On définit cet angle par rapport à la droite passant par la boule, et par l'origine, comme ceci :



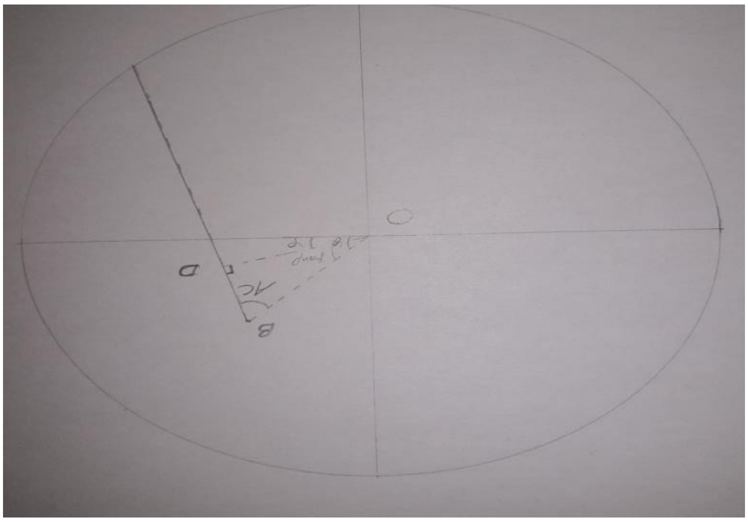
Ici θ est l'angle de la boule, et AC désigne l'angle de coup.

Ainsi on calcule les coordonnées du point (r_0, φ) de cette manière :

$r_0 = r * \sin(AC)$

Où r est la coordonnée radiale de la boule.
On notera que r_0 est constant au cours des trajectoires successives de la boule. C'est pourquoi r_0 ne dépend que des conditions initiales. On verra plus tard sur des graphes qu'en réalité, r_0 est le rayon d'un cercle par lequel la boule ne passe jamais, une caustique.

Plusieurs moyens sont possibles pour calculer φ, nous avons fait comme cela :



On définit un angle tmp, qui est l’angle DOB. Son expression est donc ; tmp = acos(r_0)/r.

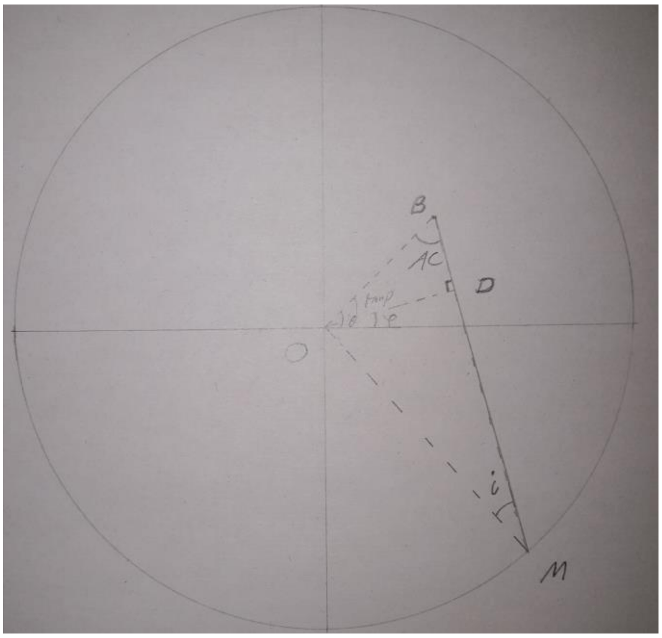
L’angle φ est alors égal à l’angle de la boule moins cet angle tmp, c’est à dire $\varphi = \theta - \text{tmp}$.

Connaissant maintenant r_0 et φ, on peut déterminer l’angle du prochain point de rebond, avec l’équation citée précédemment.

Ensuite, il faut à chaque rebond recalculer l’angle φ du nouveau point D (on a dit que r_0 ne variait pas), en utilisant l’angle du dernier rebond.

On peut aussi déterminer l’angle d’incidence i avec lequel la boule rebondit sur le cercle. Comme r_0, cet angle d’incidence ne varie pas au fur et à mesure des rebonds. Il reste le même, pour des conditions initiales données. Cela s‘explique par les lois de la réfraction, qui nous disent que l’angle de réfraction est égal à l’angle d’incidence.
On le calcule ainsi :

$i = \arcsin(\frac{r_0}{r})$



II) Aspect programmation

Pour implémenter ces équations en C, nous n'avons pas eu recours à des méthodes numériques particulières. Cependant le principal problème est le choix de coordonnée pour l'affichage sur Gnuplot.
Le système cartésien est le plus facile à utiliser sur Gnuplot, c'est pourquoi nous avions commencé par utiliser celui-ci, mais nous avons finalement changé pour les coordonnées polaires, l'utilisation du système polaire permettant une implémentation plus efficace en C, et des équations plus simples. Donc le choix du polaire est utilisé pour les équations du mouvement de la boule et une fonction a été utilisé pour faire la conversion polaire-cartésien pour la représentation graphique sur Gnuplot.

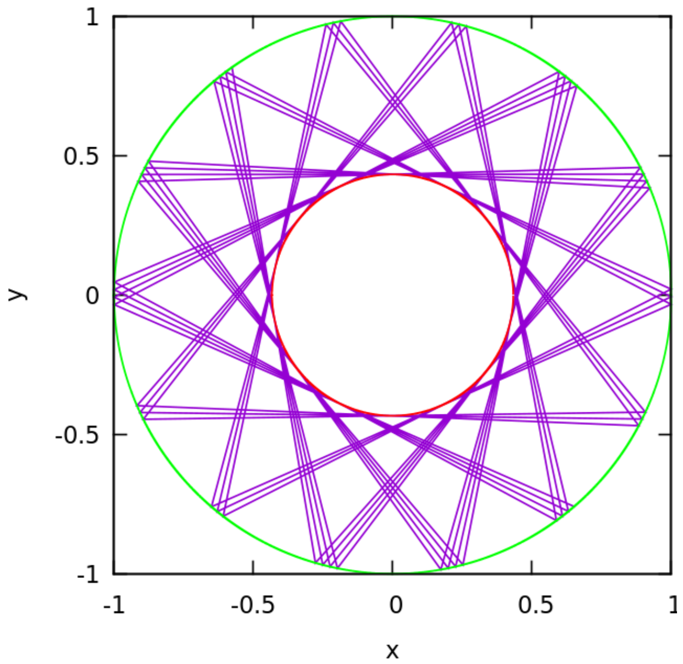
Le principe de son implémentation a été assez rapidement trouvé:

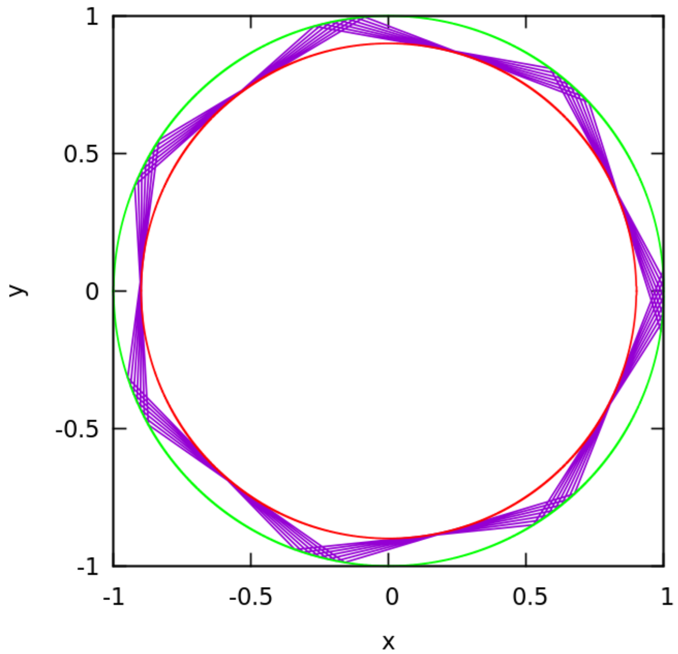
- Tout d’abord on récupère les coordonnées polaire de la boule dans le billard circulaire (pour faciliter le calcul, le rayon du billard est de 1) et l’angle du coup donné sur la boule.
- Le programme crée un second point dans la direction du coup pour créer l’équation de la droite de la trajectoire de la boule. (fonction *r_point_tangente, angle_point_tangente* et *trajectoire*).
- Comme l’angle d’incidence est par propriété toujours constant le programme le calcule(fonction *angle_incidence*).
- Ensuite on demande le nombre de rebond de la boule sur billard pour former le graphe avec le nombre de rebond comme condition d’arrêt.
- A partir de ça, la boucle commence: la valeur de l’angle point de rebond est calculé, la valeur est converti et mémorisé dans un fichier. Puis on recalcule la valeur du nouveau point pour l’équation de la droite.

Notons qu’étant donnée l’orientation que nous avons choisi pour définir l’angle de tir, nous avons du changé un signe dans l’équation donnant la position du rebond, celle-ci devenant

$\theta = -\arccos(r_0) + \varphi$ (r = 1 dans le programme, d’où la simplification de l’expression)

Les résultats obtenus nous démontrent que si on trouve le bon sens du tir de la boule en fonction de son emplacement, on voit apparaître avec un certain nombre de rebond un schéma récurrent du déplacement de la boule avec un décalage epsilon. Ce phénomène est appelé “Caustique”. Ces caustiques créent un second cercle (de rayon r_0) vide de toute trajectoire de la boule dans le billard qui varie de taille en fonction de la position de départ de la boule dans le cercle qui définit le rayon de ce cercle. En voici deux exemples :





Le cercle vert représente les limites du billard (de rayon 1), tandis que le cercle rouge représente la limite de la zone dans lequel la bille ne circule pas. En violet sont représentées les trajectoires.

Le premier : coordonnées de la boule : ($\frac{1}{2}$, $\frac{\pi}{4}$) et angle de tir = $\frac{\pi}{3}$.

Le deuxième : coordonnées de la boule : (0,9, $\frac{\pi}{6}$) et un angle de tir de $\frac{\pi}{2}$.

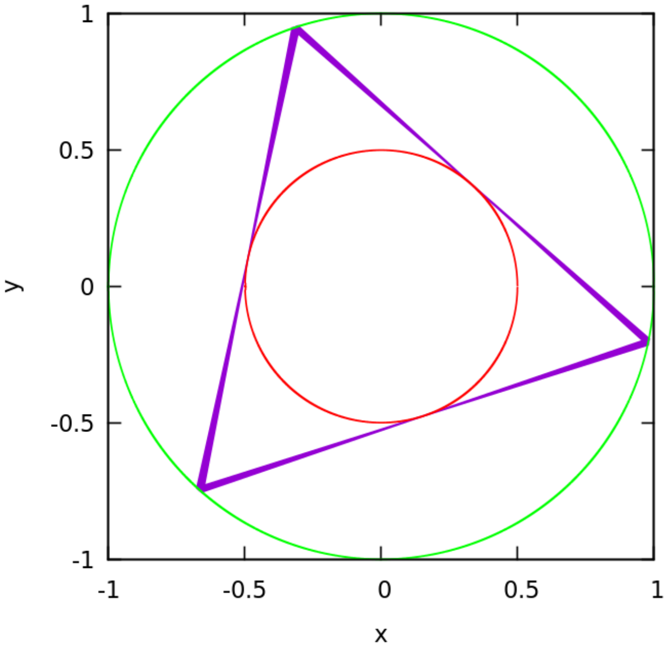
Ces deux exemples ont été traités pour une bille effectuant 50 rebonds.

Notons aussi le cas particulier où la bille passe par l'origine. Dans ce cas, la bille effectue des «allers-retour» entre les deux mêmes points, et sa trajectoire est donc un segment.

Cela n'est pas forcément évident à voir dans tous les cas, à cause de la précision machine, mais on remarque que peu importe la position de la bille au départ, les trajectoires finissent par se répéter. On a donc essayé de créer des cas où la bille va rebondir à n endroits différents du billard, puis répéter le même schéma, c'est à dire que le point de rebond n+1 est égal au premier point de rebond.
Pour cela, on fixe la boule à une position de départ. On fait ensuite varier l'angle de tir, et pour chaque angle de coup on garde en mémoire la position du premier point de rebond, puis à chaque rebond suivant, on regarde si le point est situé au même endroit que ce premier point (à un epsilon près). Si cette condition n'est pas atteinte en un certain nombre de rebond (on prend généralement 50 rebond par convention), on incrémente l'angle de tir, et on recommence les mêmes opérations.

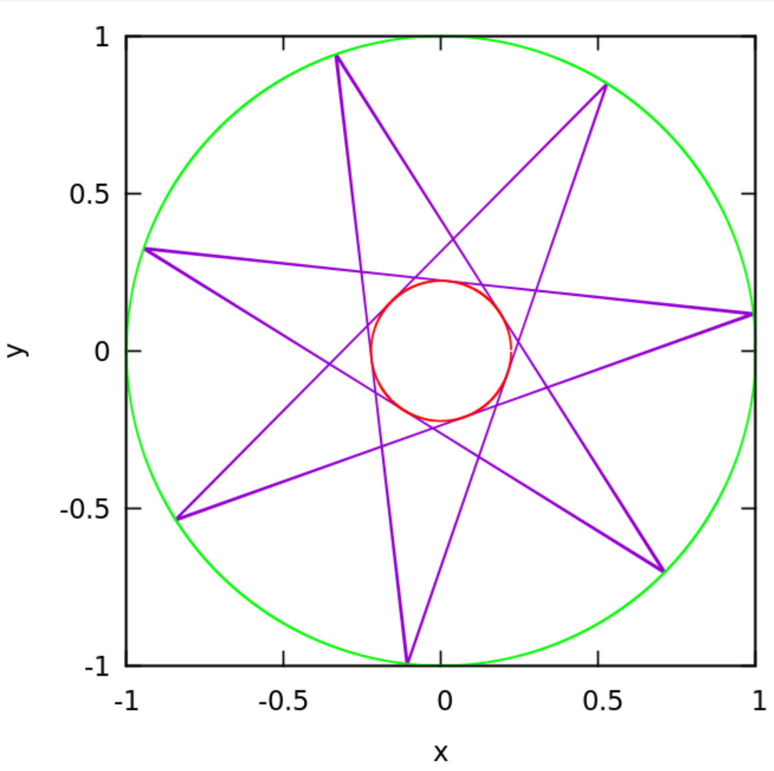
L’angle de tir varie entre 0 et 2π.

Voici des résultats obtenus par cette méthode :



Pour une boule étant positionnée au départ en $(\frac{1}{2}, \frac{\pi}{4})$.

Un autre exemple



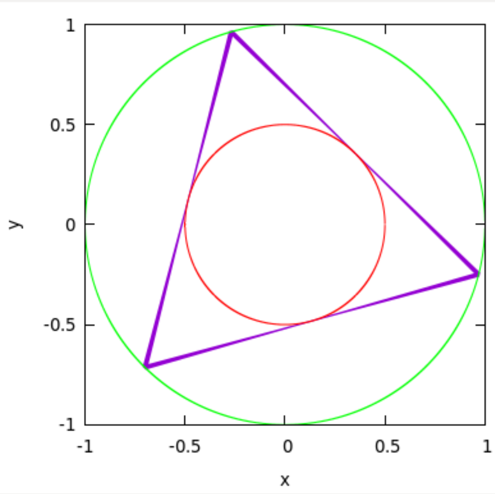
Pour une boule initialement placée en $(\frac{3}{4};\frac{\pi}{2})$

On constate que la bille ne rebondit pas exactement aux mêmes endroits qu'elle devrait, d'où « l'épaisseur » sur les graphes au niveau des points de rebond. Cela est dû au fait que lorsque nous comparons les points de rebond, on le fait à un epsilon près (0,005). Étant donné que l'on fait varier l'angle de tir d'un certain pas, si l'on ne compare les points de rebond à un epsilon près, le programme ne trouvera pas de solutions dans la plupart des cas. Nous avons tenté de mettre en équation ce problème, c'est à dire déterminer l'équation de l'angle du n-ième point de rebond, mais l'équation devient rapidement compliqué quand n croît, étant donné que l'équation dépend de son résultat pour n-1. C'est une fonction récursive. Nous avons donc préféré consacrer notre temps à la mise en forme du programme.

On constate un autre problème au niveau de certaines trajectoires recherchées.

Par exemple, en en cherchant une trajectoire de 9 rebonds différents, on obtient ceci :

:



Cela est un problème de multiple, étant donné que dans ce cas-là, le neuvième point de rebond est bien égal au premier, mais le quatrième et le septième aussi. Nous avions commencé à réfléchir ce problème, mais par manque de temps nous n'avons pas pu le régler.

En conclusion, nous avons réussi à déterminer le déplacement d'une boule dans le billard circulaire en fonction de sa position d'origine, ce qui nous a permis de découvrir un phénomène qui nous était inconnu ; les caustiques. Par ailleurs cette modélisation de la trajectoire d'une bille de billard illustre bien la sensibilité d'un système à ses conditions initiales. Avec plus de temps, nous aurions essayé de modéliser ces trajectoires dans un billard circulaire tronqué. Nous avions commencé, mais l'implémentation des calculs pour la gestion des rebonds sur le bord plan s'est avérée complexe à mettre en œuvre en parallèle avec les rebonds sur le bord circulaire. Nous avons donc préféré étudier les trajectoires qui se répètent.