TP3 – TENNIS

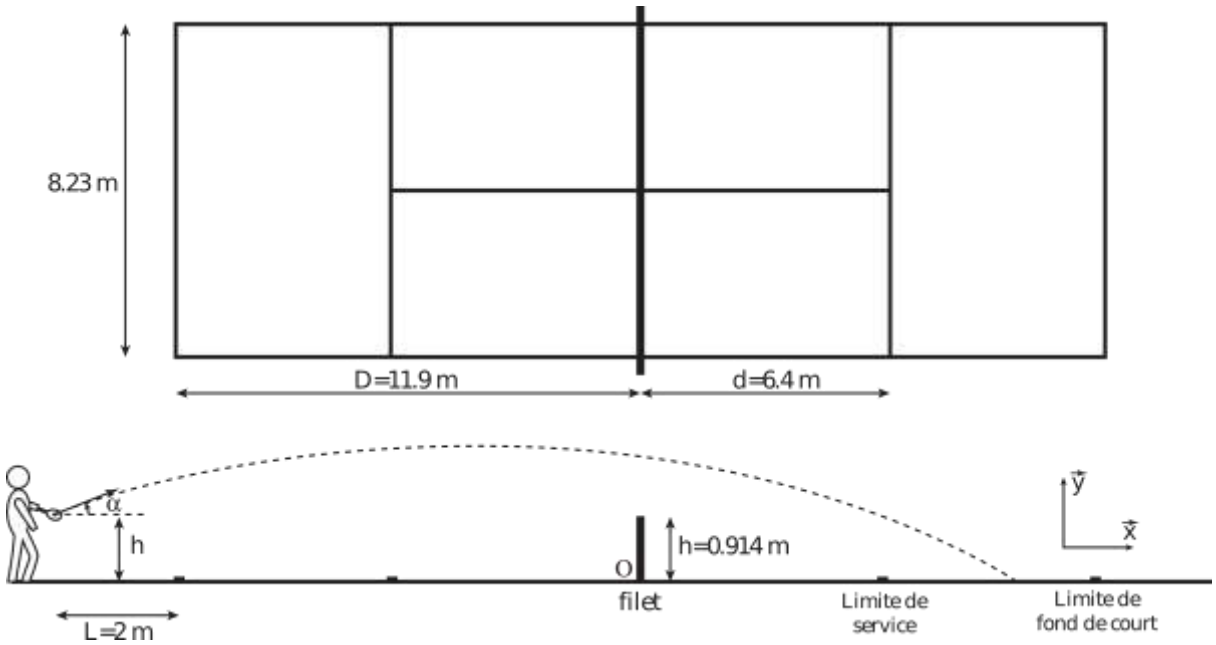
OBJECTIFS :

Le but de ce TP est de créer une petite application permettant de tracer la trajectoire d'une balle de tennis en fonction des paramètres de frappe du joueur… il s'agit surtout de coder la trajectoire et vous pourrez finaliser l'aspect GUI dans un deuxième temps.

1. PARAMETRAGE

On suppose en première approximation que la balle est renvoyée dans la direction du court de tennis. La direction est donc définie par l'angle α entre l'horizontale et la vitesse de la balle. Soit v_0 la vitesse de la balle juste après la frappe. On supposera que le joueur frappe la balle à une distance $L = 2\,m$ derrière la limite de fond de cours, à une hauteur $h = 0.914\,m$ (identique à la hauteur du filet). On note $M(x,y)$ le centre de gravité de la balle. Le centre O du repère est choisi au pied du filet.

La balle doit passer au-dessus du filet (hauteur $h = 0.914\,m$) et toucher le sol avant la limite de fond de court (distance filet-fond de court : $D = 11.9\,m$).



2. TRAJECTOIRE DE LA BALLE

2.1.Premier tracé

Nous allons définir la méthode de calcul de la trajectoire pour que celle-ci soit effectivement la trajectoire de la balle correspondant aux paramètres de frappe du joueur.

On néglige dans un premier temps le frottement de l'air sur la balle. Sachant que le joueur donne à l'instant initial (instant de la frappe) une vitesse $\overrightarrow{V_0}$ d'angle α avec l'horizontale (axe $\vec{x}$) et en appliquant le PFD à la balle, on en déduit les équations paramétriques de la trajectoire de la balle :

$$x(t) = V_0 \cos(\alpha)\,.\,t + x_0$$

$$y(t) = -\frac{g}{2}t^2 + V_0 \sin(\alpha).\,t + y_0$$

*Q1. Créer une fonction **Calcul** permettant de calculer la trajectoire parabolique de la balle.*
*Q2. Puis créer la fonction t**raceTrajectoire** en précisant dans l'appel à la fonction **Calcul** le paramètre de vitesse et d'angle puis testez votre application.*

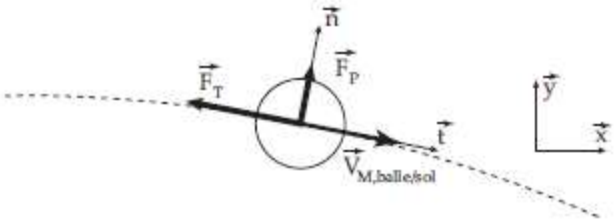
Sachant que l'on souhaite afficher 400 points sur une durée de simulation de 4s, complétez votre script en définissant les constantes ci-dessus :

- l'angle initial : $\alpha = 40°$
- le vecteur temps **t**
- le vecteur vitesse initiale bidimensionnel **Vit0** (composantes en x et en y) $v_0 = 14m.s^{-1}$
- le vecteur position initiale bidimensionnel **Pos0** (composantes en x et en y) $x_0 = -12.5m \quad ; \quad y_0 = 1m$
- les variables **x** et **y** correspondant respectivement à **x(t)** et **y(t)**
- en faisant tracer la trajectoire de la balle.

2.2. Trajectoire réelle

Hormis pour le service, la trajectoire de la balle est sensiblement perturbée par l'action de l'air. La viscosité de l'air conduit à un effort de traînée, proportionnel au carré de la vitesse : $\overrightarrow{F_T} = -\frac{1}{2}\lambda.V.\vec{V}$ où $\lambda = 26.10^{-4}N.m^{-2}.s^{-2}$. Lorsque les balles sont coupées (le joueur donne une vitesse de rotation à la balle), l'effet Magnus conduit à un effort de portance normal à la vitesse, orienté vers le haut ou vers le bas selon le sens de rotation : $\overrightarrow{F_P} = F_P.\vec{n}$ où $F_P = \mu.V.\omega$ avec une vitesse maximale de rotation de l'ordre de $\omega = \pm 100\text{rad.s}^{-1}$ et un coefficient de portance $\mu = 10^{-4}N.m^{-1}.s^{-2}$.

On notera θ l'angle $\theta = (\vec{x}(t), \vec{y}(t))$ et m la masse de la balle (m=58g)



Q3. Ecrire les équations différentielles du mouvement tenant compte de l'action de l'air sur la balle.
Vous devez obtenir les équations suivantes. Ces équations sont-elles linéaires ?

$$a_x = -0.5.\lambda.\sqrt{v_x^2 + v_y^2} * \frac{v_x}{m} \quad ; a_y = -g - 0.5.\lambda.\sqrt{v_x^2 + v_y^2} * \frac{v_y}{m}$$

Le schéma temporel adopté pour l'intégration est le suivant :

$$\begin{cases} v_{k+1} = v_k + \Delta t * a_k \\ p_{k+1} = p_k + \Delta t * v_k + \frac{1}{2}\Delta t^2 * a_k \end{cases}$$

où *a* est l'accélération (*ax ou ay*), *v* est la vitesse (*vx ou vy*) et *p* la position (*x ou y*).

La procédure d'intégration numérique consiste, à partir des conditions initiales, à appliquer récursivement
– le calcul de a_k par les équations du mouvement,
– le calcul de v_k et p_k par le schéma numérique.

On souhaite tout d'abord tenir compte de la viscosité de l'air uniquement dans le calcul de la trajectoire (pas de l'effet Magnus).

Q4. Dupliquez la partie de code correspondant au calcul du tir parabolique puis ajoutez la variable de viscosité en argument (variable booléenne indiquant si le calcul se fait avec ou sans viscosité).
Q5. Proposer alors un programme permettant de résoudre numériquement l'équation du mouvement de la balle.

La prise en compte de l'effet Magnus amène les modifications suivantes sur l'expression des accélérations :

$$a_x = \frac{-0.5.\lambda.\sqrt{v_x^2+v_y^2} * v_x - \mu.v_y.\omega}{m} \quad ; \quad a_y = -g * \frac{-0.5.\lambda.\sqrt{v_x^2+v_y^2} * v_y + \mu.v_x.\omega}{m}$$

Q6. Proposer enfin un autre programme permettant de tenir compte de l'effet Magnus puis tracer.

2.3. Prise en compte du rebond

Jusqu'à présent, le calcul se fait sans tenir compte du rebond de la balle sur le sol.

Une modélisation mathématique du rebond peut revenir simplement à changer le signe de la composante en y de la vitesse v_y et à l'atténuer d'un facteur *e* (coefficient de restitution au rebond) : $v_y = -e * v_y$. Encore faut-il détecter le rebond ?!...

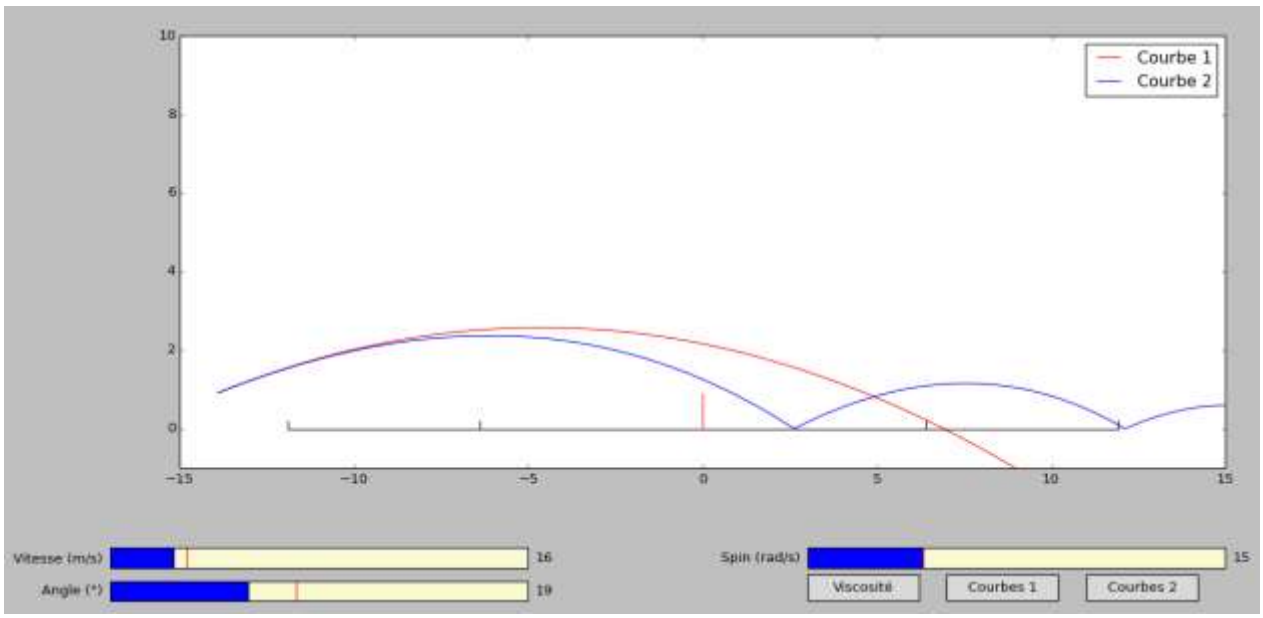
Q7. Expliquez comment détecter un rebond et modifiez votre script afin de prendre en compte le rebond et l'atténuation de la vitesse qui en résulte.

Q8. D'autre phénomène peuvent être pris en compte comme l'incidence du spin sur le rebond, l'effet du vent sur la trajectoire. Vous pouvez aussi vous lancer dans le calcul d'une trajectoire en trois dimensions...

3. GRAPHISMES

Le début du programme est constitué de trois fonctions, une fonction Trace_terrain() permettant de tracer le terrain de tennis, et deux fonctions plot_courbe1() et plot_courbe2(), renvoyant le résultat r=[x,y]; contenant les deux les abscisses x et ordonnées y de la trajectoire en fonction du temps t pour deux courbes rouge et bleu.

La seconde partie du programme relève de la construction de l'interface graphique :



4.1. Terrain

Nous commencerons par tracer le terrain de tennis dans la fenêtre de l'application. Pour cela on choisit de représenter le terrain vu de profil (image du bas sur la première figure) … la 3D c'est pour plus tard !!

Ouvrez le fichier *Graphique_eleve.py* dans python. Vous trouverez la déclaration de l'interface graphique avec les importations de widgets nécessaires, les paramètres de l'étude et quatre fonctions principales plus 3 petites fonctions de changement de variable globale (*chgt*).

La première fonction permet de tracer le terrain dans la fenêtre graphique et *courbe 1* et *courbe 2* permettent de créer la trajectoire.

La dernière fonction (*update_figure(val)*) est sans doute la plus importante : elle est lancée à chaque rafraichissement de la fenêtre graphique. On y trouve un appel aux fonctions *courbe1()* et *courbe2()* permettant de tracer respectivement les trajectoires de la balle.

Nous nous intéressons maintenant à la fonction *TraceTerrain()*.

Celle-ci est à peine commencée, pour vous donner les outils de base : la largeur et la hauteur de la fenêtre graphique (exprimées en pixels) sont placées dans les variables *largeur* et *hauteur*. Une commande permet de définir la couleur du tracé (le noir) et le rouge pour le filet. On remarquera que le système d'axe a pour origine le coin en haut à gauche et l'axe x est orienté vers la droite tandis que y est orienté vers le haut

Q9.A vous de réaliser votre procédure permettant de tracer le terrain de tennis en haut de la fenêtre graphique. Je vous invite à utiliser un repère intermédiaire dont l'origine est au pied du filet, l'axe x vers la droite et l'axe y vers le haut, et dont les dimensions sont exprimées en mètre pour faciliter par la suite le tracé de la trajectoire.

Q10. Testez votre application en lançant le programme.

Le terrain de tennis doit se dessiner correctement, de manière analogue à la fenêtre proposée en sur la figure page 3

4.2.Trajectoire de la balle

Nous allons maintenant nous occuper de tracer la trajectoire de la balle sur la fenêtre graphique.

Les paramètres sont renseignés dans les fonctions *courbe1* et *courbe 2*. Il vous suffit de compléter les parties manquantes (A COMPLETER) avec les programmes mis en place dans la partie 2.1 et 2.2. N'oubliez pas de renseigner le vecteur vitesse initiale.

La fonction *courbe 1* renvoie :

- ***X** contenant la liste des abscisses de la trajectoire pour une durée t (**vecteur temps créé à la ligne 98)***
- ***Y** contenant la liste des ordonnées de la trajectoire pour une durée t*

La fonction *courbe 2* renvoie :

- ***pxr** contenant la liste des abscisses de la trajectoire pour une durée t*
- ***pyr** contenant la liste des ordonnées de la trajectoire pour une durée t*

4.3.Sliders et Boutons

Pour aller plus vite, j'ai utilisé ici les widgets associés à Matplotlib plutôt que de faire une interface Tkinter, mais cette solution est aussi envisageable et ressemblerait à la proposition que vous allez coder ici.

*Q11. En vous inspirant du slider 1 Vitesse, réaliser les 2 autres **sliders : Angle et Spin**. Bien les positionner par rapport à la figure page 3*

*Q12. En vous inspirant du bouton Courbe 1, réaliser les **2 autres boutons : Viscosité et Courbe 2**. Il faudra alors compléter les fonctions **chgt2** et **chgtvis** pour affecter les valeurs c2 et visc lors d'un click … S'inspirer de chgt1.*

4. POUR ALLER PLUS LOIN … LE RETOUR

On choisit maintenant de ne garder que la courbe 2 (la plus réelle) imaginer une solution pour tracer la trajectoire retour :

- *En déterminant la valeur de y sur le bord droit du terrain*
- *En mettant des sliders pour le joueur de droite (vitesse,angle,spin)*
- *En traçant la trajectoire de retour en vert.*