Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет систем управления и робототехники

Отчет по проектной работе

по теме "Движение вращающегося мяча в воздухе с учетом эффекта Магнуса"

по дисциплине "Механика"

Выполнили: студенты

Нечаева А.А.
Попов В.?.

Преподаватель: *Смирнов Александр Витальевич*

Санкт-Петербург, 2023

1 Построение математической модели

1.1 Эффект Магнуса

Эффект Магнуса - физическое явление, возникающее при обтекании вращающегося тела потоком жидкости или газа. Возникающая сила - результат воздействия таких физических явлений, как *эффект Бернулли* и образования *пограничного слоя* в среде вокруг обтекаемого объекта, - действует на тело перпендикулярно направлению потока. Эффект описан немецким физиком *Генрихом Магнусом* в 1853 году.

Вращающийся объект создает вокруг себя вихревое движение, с одной стороны направление вихря совпадает с направлением обтекающего потока, с другой - противоположно, следовательно, скорость движения среды с одной стороны повышается, с другой - уменьшается. *Согласно уравнению Бернулл: чем меньше скорость, тем выше давление.* Возникающая разность давлений вызывает возникновение поперечной силы, вектор которой направлен от стороны, где направления вращения и потока противоположны, стороне с сонаправленными. Явление иллюстрирует рисунок 1.

рисунок 1

1.2 Вывод уравнений и формул

Обозначим начальные условия полета мяча:
1. Мяч - *абсолютно твёрдое тело*, то есть будем считать, что *взаимное расположение точек мяча не меняется с течением времени*, деформацией во время полета пренебрежем
2. Известные параметры мяча: радиус ($\boldsymbol{R}$) и масса ($\boldsymbol{m}$)
3. Также заданы начальные значения угловой ($\omega_{\mathbf{0}}$) и линейной скоростей ($\mathbf{v_0}$)
4. Задана некоторая плотность газа - среды, в которой происходит полет мяча (ρ)

Запишем *закон Ньютона* в общем виде

$$\vec{F} = m \cdot \vec{a} \tag{1}$$

Далее распишем силы, действующие на мяч в процессе полета

$$\vec{F}_{\text{тяжести}} + \vec{F}_{\text{Магнуса}} = m \cdot \vec{a} \tag{2}$$

Пусть шар находится в потоке набегающего не него идеального газа. Скорость потока на бесконечности $\vec{u}_\infty$. Чтобы сымитировать вращение шара, введем циркуляцию скорости Γ вокруг него. Исходя из закона Бернулли, можно получить, что полная сила, действующая в таком случае на шар, равна:

$$\vec{R} = -\rho\vec{\Gamma} \times \vec{u}_\infty \,, \tag{3}$$

где $\vec{u}_\infty = -\vec{v}$ (v – линейная скорость мяча); Γ вычислим как

$$\Gamma = \oint\limits_L v_\tau dS \,, \tag{4}$$

v_τ – проекция скорости на касательную к этой кривой, dS – элемент длины кривой. В случае шара запишем Запишем формулу для вычисления силы Магнуса в общем виде:

$$\Gamma = \int\limits_{-R}^{R} 2\pi\omega(R^2 - x^2)\, dx = 2\pi\omega \left(R^2 x - \frac{x^3}{3} \right)\Bigg|_{-R}^{R} =$$
$$= 2\pi\omega \left(R^3 - \frac{R^3}{3} + R^3 - \frac{R^3}{3} \right) = 4\pi\omega \frac{2R^3}{3} = \frac{8\pi\omega}{3} R^3 \,, \tag{5}$$

где R – радиус шара, ω – заданная угловая скорость вращения мяча. Будем считать, что вектор угловой скорости задан вдоль 1 оси – оси Z.
Тогда запишем формулу для вычисления силы Магнуса:

$$\vec{F}_{\text{Магнуса}} = -\rho\vec{\Gamma} \times \vec{u}_\infty = -\rho \frac{8\pi\vec{\omega}}{3} R^3 \times -\vec{v} = \frac{8\pi}{3} \rho R^3 \vec{\omega} \times \vec{v} \tag{6}$$

В общем случае будем рассматривать движение мяча в *Декартовой системе координат*, в *трехмерном пространстве*, тогда проекции на оси X, Y и Z соотвественно

$$\begin{cases} m\ddot{x} = \frac{8\pi}{3} \rho R^3 \left(\vec{\omega} \times \vec{v} \right)_x \,, \\ m\ddot{y} = \frac{8\pi}{3} \rho R^3 \left(\vec{\omega} \times \vec{v} \right)_y \,, \\ \ddot{z} = g \end{cases} \tag{7}$$

Перейдем к уравнениям для угловой скорости вращения мяча ω, в частности, найдем выражения для разложения $(\vec{\omega} \times \vec{v})$ по единичным векторам, задающим оси координат, $\vec{i}$, $\vec{j}$ и $\vec{k}$

$$\begin{gathered}
(\vec{\omega} \times \vec{v}) = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ \omega_x & \omega_y & \omega_z \\ v_x & v_y & v_z \end{vmatrix} = \vec{i} \begin{vmatrix} \omega_y & \omega_z \\ v_y & v_z \end{vmatrix} - \vec{j} \begin{vmatrix} \omega_x & \omega_z \\ v_x & v_z \end{vmatrix} + \vec{k} \begin{vmatrix} \omega_x & \omega_y \\ v_x & v_y \end{vmatrix} = \\
= \vec{i}\,(\omega_y v_z - \omega_z v_y) + \vec{j}\,(\omega_z v_x - \omega_x v_z) + \vec{k}\,(\omega_x v_y - \omega_y v_x) \rightarrow \\
\rightarrow (\vec{\omega} \times \vec{v})_x = \omega_y v_z - \omega_z v_y = -\omega_z v_y \\
(\vec{\omega} \times \vec{v})_y = \omega_z v_x - \omega_x v_z = \omega_z v_x \\
(\vec{\omega} \times \vec{v})_z = \omega_x v_y - \omega_y v_x = 0
\end{gathered} \tag{8}$$

Перейдем к дифференциальному виду уравнений проекций движения мяча на оси координат:

$$\begin{cases} m\ddot{x} = -\frac{8\pi}{3}\rho R^3 \omega_z v_y\,, \\ m\ddot{y} = \frac{8\pi}{3}\rho R^3 \omega_z v_x\,, \\ \ddot{z} = g \end{cases} \tag{9}$$

Запишем также начальные условия задачи, координаты мяча – положение его центра масс во времени

$$\begin{cases} (x(0), y(0), z(0)) = (0, 0, 0) \\ v_x(0) = v_{0x} \\ v_y(0) = v_{0y} \\ v_z(0) = v_{0z} \\ \omega_x(0) = 0 \\ \omega_y(0) = 0 \\ \omega_z(0) = \omega_0 \end{cases} \tag{10}$$

1.3 Решение дифференциальных уравнений

Для решения соотвествующей системы дифференциальных уравнений был применен *метод Эйлера* численного решения дифференциальных уравнений. Описание метода:
Пусть дана задача Коши для уравнения первого порядка:

$$\frac{dy}{dx} = f(x, y),\ y|_{x=x_0} = y_0\,,$$

где функция f определена на некоторой области $D \in R^2$. Решение ищется на полуинтервале $(x_0, b]$. На этом промежутке введем узлы $x_0 < x_1 < ... < x_n \leq b$. Приближенное решение в узлах x_i, которое обозначим через y_i, определяется по формуле

$$y_i = y_{i-1} + (x_i - x_{i-1}) f(x_{i-1}, y_{i-1}),\ \ i = 1, 2, 3, ..., n.$$

Эти формулы непосредственно обобщаются на случай систем обыкновенных дифференциальных уравнений.

В данной работе для нахождения численного решения системы линейных однородных уравнений второго порядка метод Эйлера был применен последовательно: сначала для вычисления первой производной, а затем на основе полученного результата – второй производной.

1.4 Численный эксперимент. Огибание препятствия заданной ширины

Молуль силы, действующей на мяч, постоянен, а ее вектор перпендикулярен скорости мяча в любой момент времени и лежит в плоскости XY, следовательно, траектория мяча представляет собой фрагмент окружности в плоскости XY.

Запишем центростремительное ускорение:

$$a_n = \frac{F}{m} = \frac{8\pi}{3m} \rho \omega v R^3 \tag{11}$$

2 Моделирование процесса

2.1 Общие сведения

Исходные данные программы:

1. Масса мяча (по футбольным стандартам) **m = 0.450 кг**
2. Радиус мяча (при длине окружности около 70 см) **R = 0.11 м**
3. Плотность воздуха (при $T = 15C^{o}$) $\rho = 1.225 \frac{\text{кг}}{\text{м}^3}$
4. Координаты начальной точки запуска мяча $(0, 0, 0)$

Данные вводимые пользователем:

1. Проекции линейной скорости на оси x, y, z, *ограничение:* $v_{z_0} > 0$
2. Значение угловой скорости ω, *примечание: по умолчанию вектор угловой скорости направлен вдось оси z*

3*. Задание ширины препятствия(?)

Выходные данные:
1. Для первой части эксперимента с заданными линейной и угловой скоростями выводится графическое представление полета на плоскости xy
2. Для эксперимента с заданным препятствием выводятся подходящие значения угловой и линейных скоростей, которые необходимо задать, чтобы обойти препятствие

2.2 Написание программы

Программа написана на языке *Python 3* с использованием библиотеки *Matplotlib*.

```
# -*- coding: utf-8 -*-
import matplotlib.pyplot as plt
from math import *
from mpl_toolkits.mplot3d import Axes3D
import copy

v_x = 10 *0.29
v_y =5 * 0
v = sqrt(v_x**2+v_y**2)
v_z = 10

C = 0.51    # lift coefficient (0.45 for foot) (0.51 for tennis)
R = 0.0325        # sphere radius (m) (0.11 for foot) (0.0325 for t
m = 0.055        # ball mass (kg) (0.430 for foot) (0.055 for tenn
p = 1.225       # air density (kg/m^3)

w_z = 50 * 3  # ball speed angular along the z-axis
alpha = pi/2        # Angle between v and w
w_x = w_y = 0.0
g = 9.81


dt = 0.001
lT=[0]
lX=[0]
lXPrim=[v_x]
```

```
lZ=[0]
lZPrim=[v_z]
lY=[0]
lYPrim=[v_y]



def x_second(x_first, y_first, z_first, C, R, p, v, w_z, alpha):
    return (8 / 3) * pi * (R**3) * p * sin(alpha) * (w_y * z_firs


def y_second(x_first,y_first, z_first, C, R, p, v, w_z, alpha):
    return (8 / 3) * pi * (R**3) * p * sin(alpha) * (w_z * x_firs


def z_second(x_first, y_first, z_first, C, R, p, v, w_z, m, g, al
    return -g


def XprimSuiv(x_first,XSecond, dt):
    return x_first + dt * XSecond

def YprimSuiv(y_first,YSecond, dt):
    return y_first + dt * YSecond

def ZprimSuiv(z_first,ZSecond, dt):
        return z_first + dt * ZSecond

def XSuiv(X,x_first, dt):
    return X + dt * x_first

def YSuiv(Y,y_first, dt):
    return Y + dt * y_first

def ZSuiv(Z,z_first, dt):
    return Z + dt * z_first

def tSuiv(t,dt):
    return t + dt
```

```
def constructor(lT,lX,lXPrim,lY,lYPrim,lZ,lZPrim,dt,C, R, p, v, w
    # temp = 10000
    while lZ[-1] >= 0 :
    #while temp >= 0 :
        #temp -= 1
        lT.append(tSuiv(lT[-1],dt))
        lX.append(XSuiv(lX[-1],lXPrim[-1],dt))
        lY.append(YSuiv(lY[-1],lYPrim[-1],dt))
        lZ.append(ZSuiv(lZ[-1],lZPrim[-1],dt))
        lXPrim.append(XprimSuiv(lXPrim[-1],x_second(lXPrim[-1],lY
        lYPrim.append(YprimSuiv(lYPrim[-1],y_second(lXPrim[-1],lY
        lZPrim.append(ZprimSuiv(lZPrim[-1],z_second(lXPrim[-1],lY
    return lT, lX, lXPrim,lY,lYPrim, lZ, lZPrim


Tuple = constructor(lT,lX,lXPrim,lY,lYPrim,lZ,lZPrim,dt,C, R, p,
plt.axis([-2,20,-1,25])
# add labels to the axes and title of the chart
plt.xlabel('x, m', fontsize=16)
plt.ylabel('y, m', fontsize=16)
plt.title('Projection of the flight path of the ball in the XY pl
plt.plot(Tuple[1],Tuple[3])
'''c = plt.Circle((2, 3), radius=1)
plt.gca().add_artist(c)
'''
plt.grid(which='major')
# enable additional grid
plt.grid(which='minor', linestyle=':')
plt.tight_layout()
plt.show()

plt.axis([0,20,-2,10])
plt.xlabel('x, m', fontsize=16)
plt.ylabel('z, m', fontsize=16)
plt.title('Projection of the flight path of the ball in the XZ pl
plt.plot(Tuple[1],Tuple[5])
plt.grid(which='both')
# enable additional grid
plt.grid(which='minor', linestyle=':')
plt.tight_layout()
```

```
plt.show()

plt.axis([0,20,-2,10])
plt.xlabel('y, m', fontsize=16)
plt.ylabel('z, m', fontsize=16)
plt.title('Projection of the flight path of the ball in the YZ pl
plt.plot(Tuple[3],Tuple[5])
plt.grid(which='major')
# enable additional grid
plt.grid(which='minor', linestyle=':')
plt.tight_layout()
plt.show()

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot(Tuple[1], Tuple[3], Tuple[5], label='parametric curve')

def finder(x_fst, y_fst, r_barier, R_sph, p, m, vel, g):
    Yo = (x_fst**2 + y_fst**2 -  r_barier**2)/((y_fst - r_barier)
    Deg = (8 * pi *  R_sph**3 * p * Yo)/ (3 * m)
    KOF = (2 * Yo * y_fst)/(x_fst**2 + y_fst**2)
    S = Yo * 2 * asin((KOF * sqrt(x_fst**2 + y_fst**2))/(2 * Yo))
    Vmin = (S * g) / (2 * vel)
    Wvmin = Vmin/ Deg
    return Yo, Deg, KOF, S, Vmin, Wvmin
```