Chaotic Magnetic Pendulum

Yiwei Yu
Nicolas Albiani

Problem

Consider a pendulum consisting of a magnetic bob attached to a string. If the pendulum is allowed to swing over a structure of permanent magnets, it will display complex motion.

Study the pendulum dynamics and its dependence on the number of permanent magnets and their arrangements

— International Physicist Tournament 2021

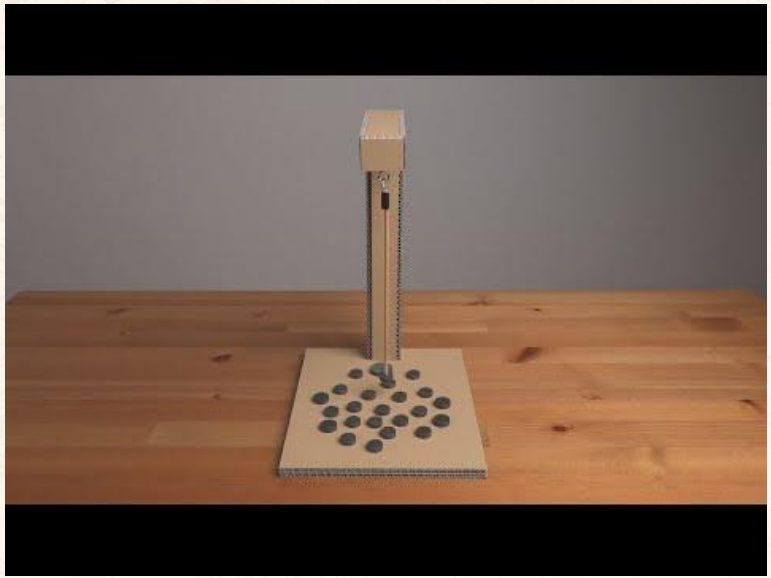


TABLE OF CONTENTS

01 Simulation Set Up

02 Variable Conditions

03 Animation

04 Basins of attraction

Simulation Set Up

01

1. Assumptions in 2D simulation
2. Set up magnets and forces
3. Path Generation

1. Assumptions in 2D Simulation

Small Angle
- Linear Restore
- Static Z

Monopole Magnets
- Like Charges

2. Set up magnets and forces

Force Model
- Componentwise
- Arrays

Static Base Forces
- Drag
- Gravity

```
def aY(x,y,vy):

  #Contribution from gravity
    ygrav = -K*y

  #Contribution from drag
    ydrag = -B*vy


  #Contribution from magnet one
    if y-MY[0] == 0:
        ymag1=0
    else:
        ymag1 = sign(MY[0]-y)*(J*S[0]*(abs(np.cos(np.arctan((x-MX[0])/(y-MY[0])))))/((x-MX[0])**2+(y-MY[0])**2+(Z)**2)**1.5
  #Contribution from magnet two
    if y-MY[1] == 0:
        ymag2=0
    else:
        ymag2 = sign(MY[1]-y)*(J*S[1]*(abs(np.cos(np.arctan((x-MX[1])/(y-MY[1])))))/((x-MX[1])**2+(y-MY[1])**2+(Z)**2)**1.5
  #Contribution from magnet three
    if y-MY[2] == 0:
        ymag3=0
    else:
        ymag3 = sign(MY[2]-y)*(J*S[2]*(abs(np.cos(np.arctan((x-MX[2])/(y-MY[2])))))/((x-MX[2])**2+(y-MY[2])**2+(Z)**2)**1.5

    return ygrav + ydrag + ymag1 + ymag2 + ymag3
```

3. Path Generation - Initial Conditions

- Set up empty arrays
  - N = Number of steps
- Insert initial conditions
  - Index 0

```
#Declare main arrays
t = np.empty(N) #Time array
x = np.empty(N) #x position array
y = np.empty(N) #y position array
vx = np.empty(N) #x velocity array
vy = np.empty(N) #y velocity array
```

```
#Initial conditions (start with no velocity)
t[0] = 0 #Initial time (in sec)
x[0] = 5 #Initial x position (in m)
y[0] = 1 #Initial y position (in m)
vx[0] = 0 #Initial x velocity (in m/s)
vy[0] = 0 #Initial y velocity (in m/s)
```

3. Path Generation - Loop

- Loop through N steps
  - i = index for each array
  - Update x, y values for every i
  - Use the equations of kinematics

```
for i in range(N-1):
        t[i+1] = t[i] + DT

        x[i+1] = x[i]+vx[i]*DT+(aX(x[i],y[i],vx[i])*(DT**2)*0.5)
        y[i+1] = x[i]+vy[i]*DT+(aY(x[i],y[i],vy[i])*(DT**2)*0.5)

        vx[i+1] = vx[i] + aX(x[i],y[i],vx[i])*DT
        vy[i+1] = vy[i] + aY(x[i],y[i],vy[i])*DT
```

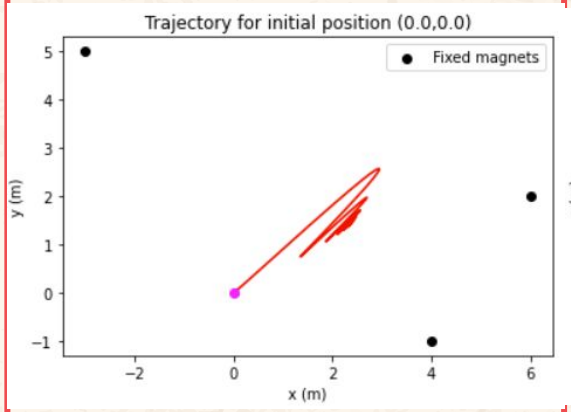
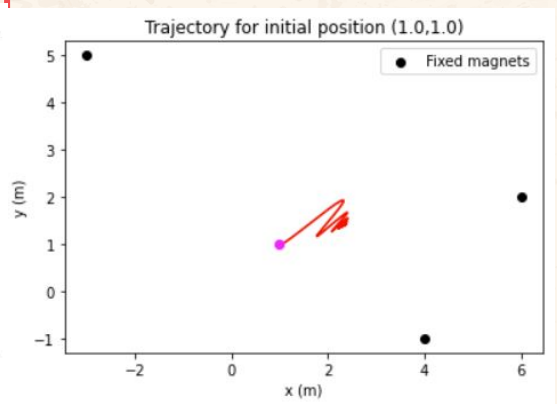
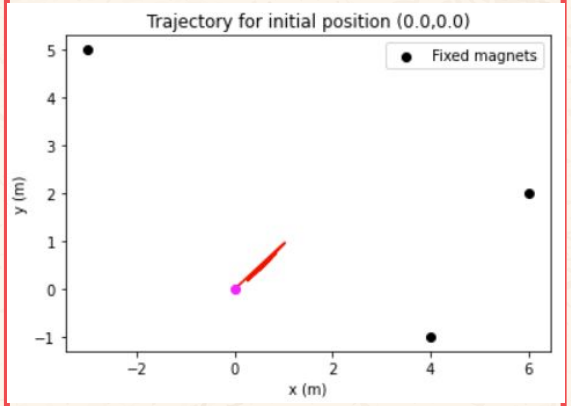
```
#Plot the trajectory
plt.plot(x,y,zorder=1,color='red')
plt.scatter(MX,MY,color='black',zorder=2,label='Fixed magnets')
plt.scatter(x[0],y[0],color='magenta',zorder=3)
title = 'Trajectory for initial position ('+str(x[0])+','+str(y[0])+')'
plt.title(title)
plt.xlabel('x (m)')
plt.ylabel('y (m)')
plt.legend()
plt.show()
```

Varying Conditions

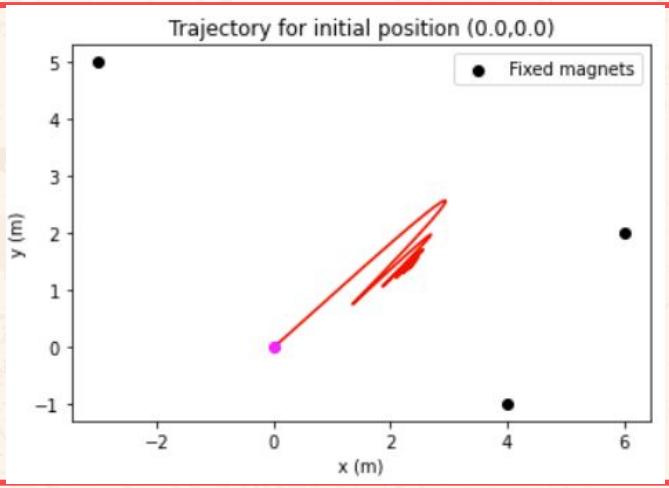
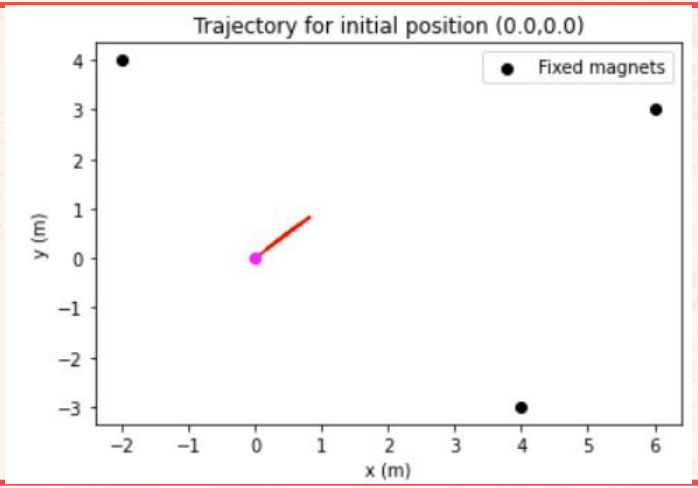
02

1. Initial Position
2. Positions of magnets
3. Strengths of magnets

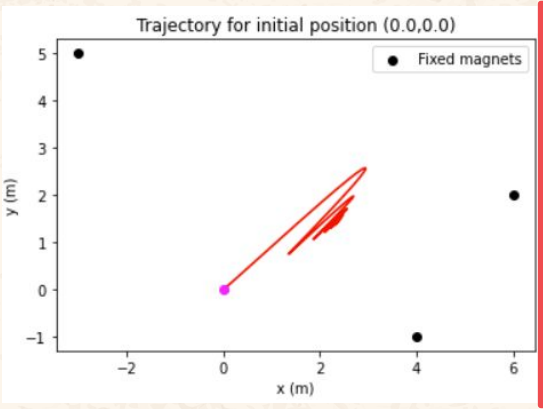
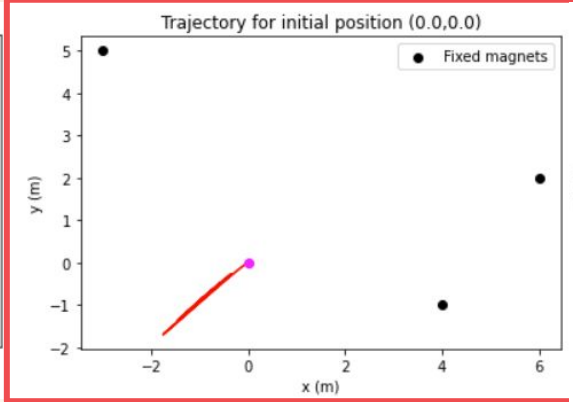
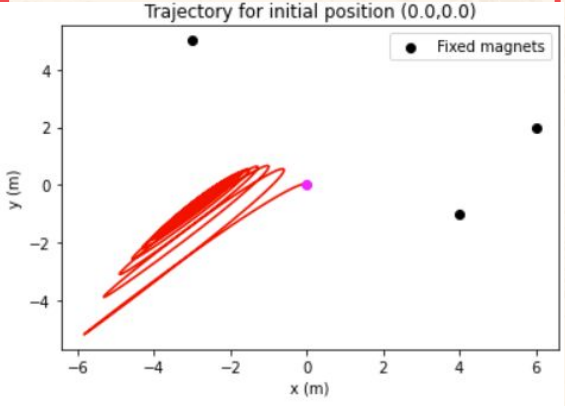
Initial Position

(0, 0, .25)	(1, 1, .25)	(0, 0, 5)
		

Positions of magnets

(-3,5/4,-1/6,2)	(4,-3/6,3/-2,4)
	

Strengths of magnets

(10, 10,10)	(-10, -10, -10)	(100, -40, -40)
		

Animation

- Loop through i to get a set of frames

```
# Initilize writer
metadata = dict(title='2D animation', artist='Matplotlib')
writer = FFMpegWriter(fps=50, metadata=metadata, bitrate=200000)
fig = plt.figure(dpi=200)

fig, ax = plt.subplots()

## SAVE AS MP4 ##
with writer.saving(fig, "magnet.mp4", dpi=200):
    for i in range(N-1):
        ax.clear()

        ax.plot(x,y,zorder=1,color='magenta')
        ax.scatter(MX,MY,color='black',zorder=2,label='Fixed magnets')
        ax.scatter(x[i],y[i],color='magenta',zorder=5)

        plt.draw()
        plt.pause(0.00001)
        writer.grab_frame()
```

Line of path

Permanent magnets

Pendulum bob

03

Animation

An interactive illustration of how the pendulum swings

Animation

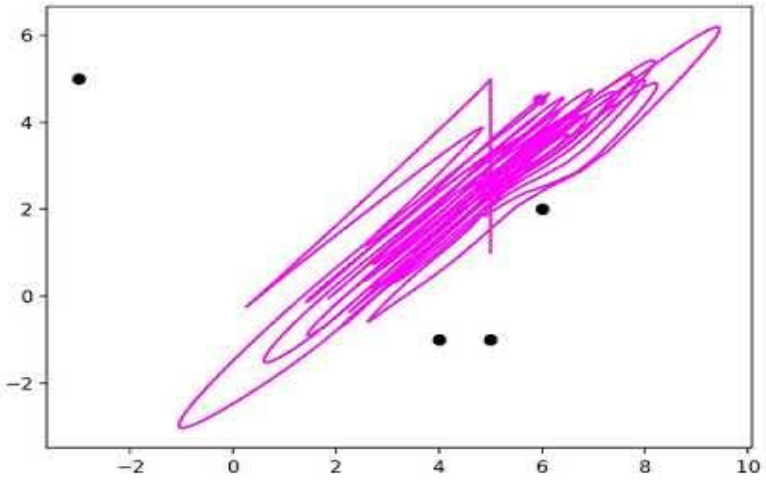
- Loop through i to get a set of frames

```
# Initilize writer
metadata = dict(title='2D animation', artist='Matplotlib')
writer = FFMpegWriter(fps=50, metadata=metadata, bitrate=200000)
fig = plt.figure(dpi=200)

fig, ax = plt.subplots()

## SAVE AS MP4 ##
with writer.saving(fig, "magnet.mp4", dpi=200):
    for i in range(N-1):
        ax.clear()

        ax.plot(x,y,zorder=1,color='magenta')
        ax.scatter(MX,MY,color='black',zorder=2,label='Fixed magnets')
        ax.scatter(x[i],y[i],color='magenta',zorder=5)

        plt.draw()
        plt.pause(0.00001)
        writer.grab_frame()
```

Line of path

Permanent magnets

Pendulum bob



Demonstration

Path above 4 magnets

Pendulum: (5, 1, 0)

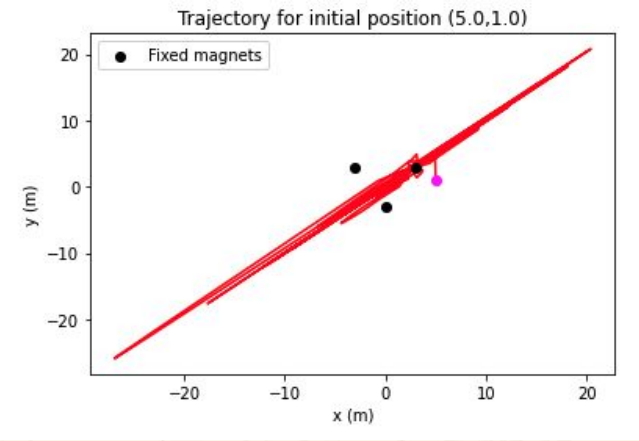
Magnets:(-3, 5, 0) (4, -1, 0) (6, 2, 0) (5, -1 , 0)

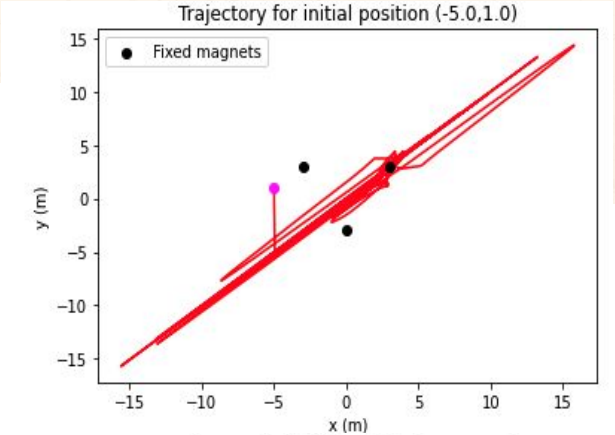
03

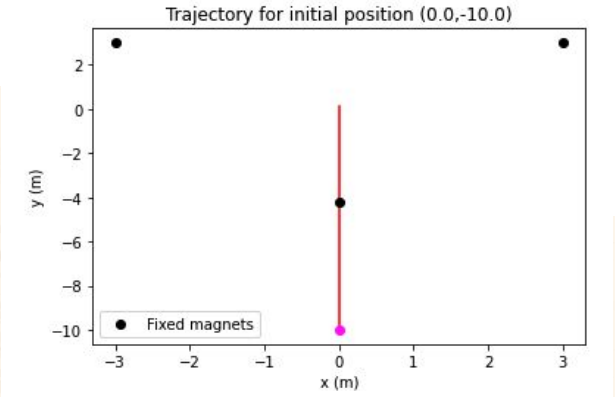
Basins of attraction

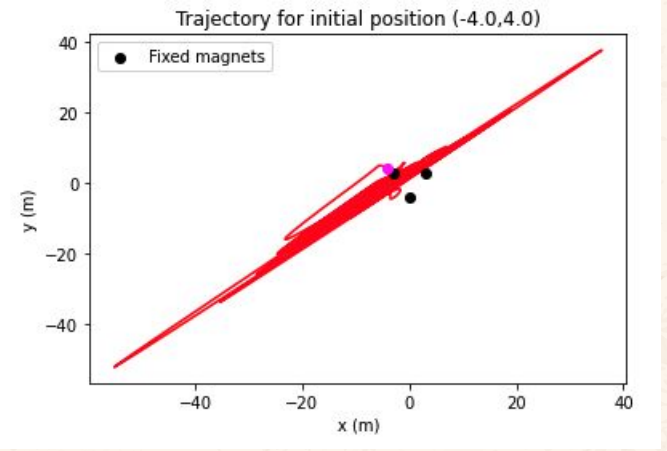
Regular patterns in chaotic behaviour

Patterns observed with special magnet positions

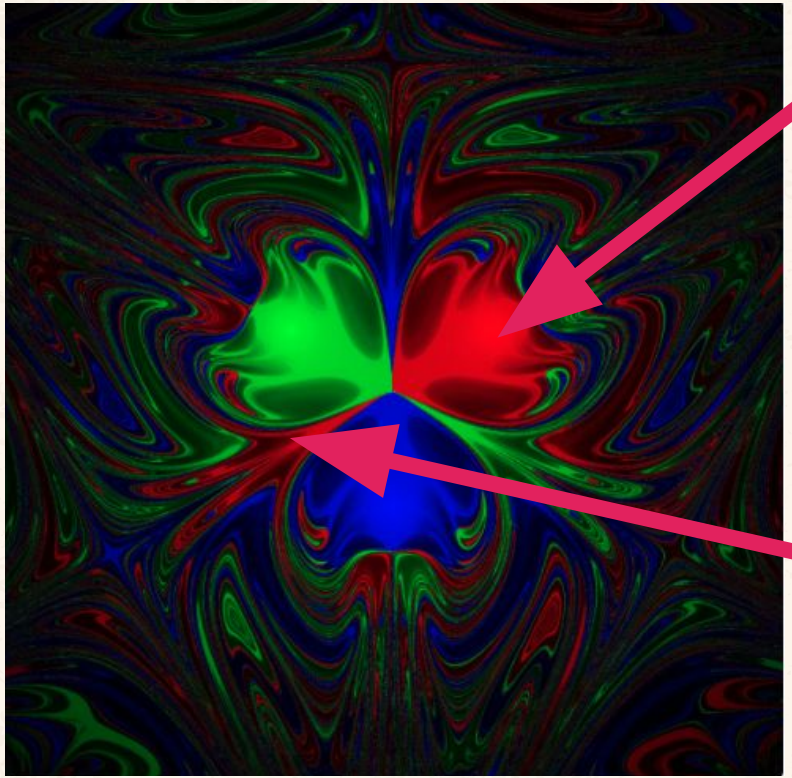


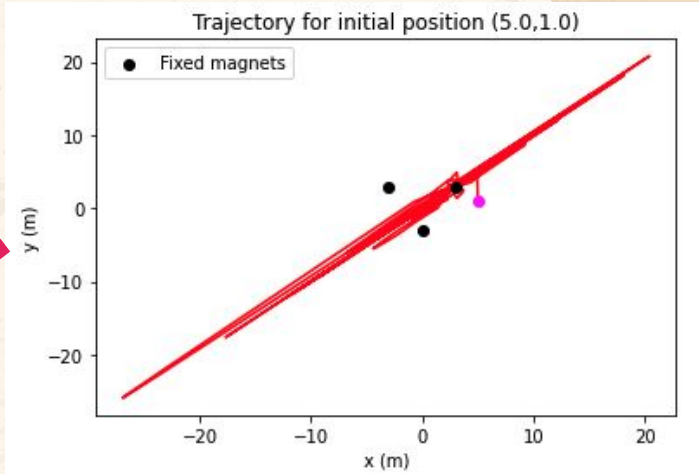


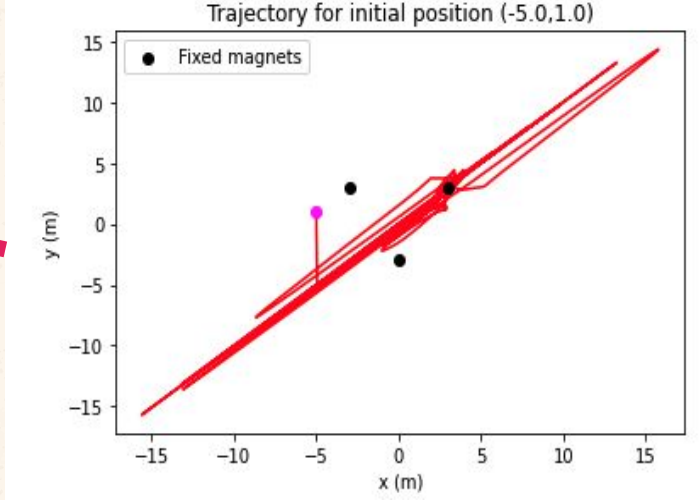




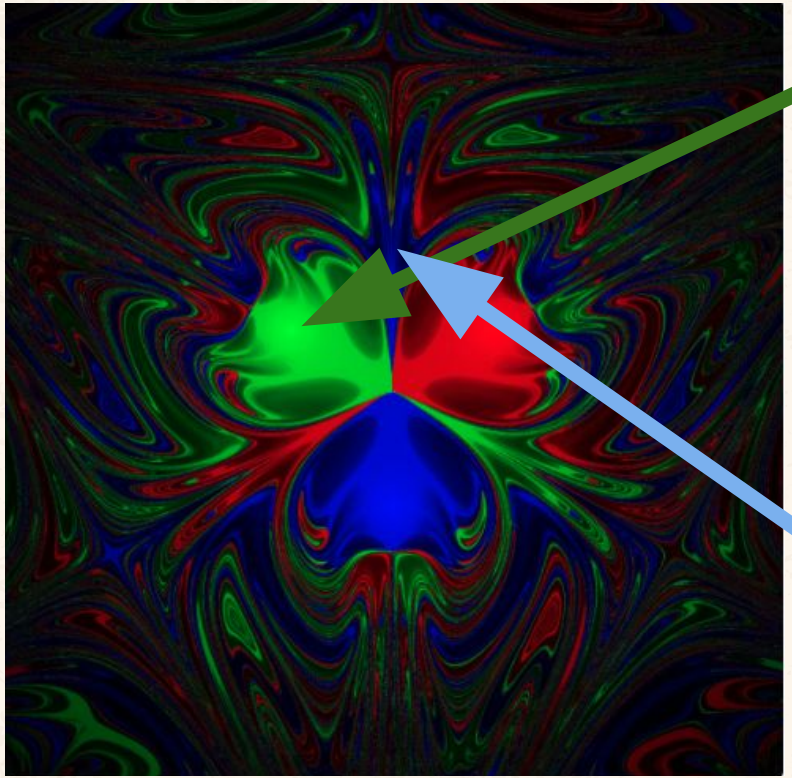
Basins of attraction

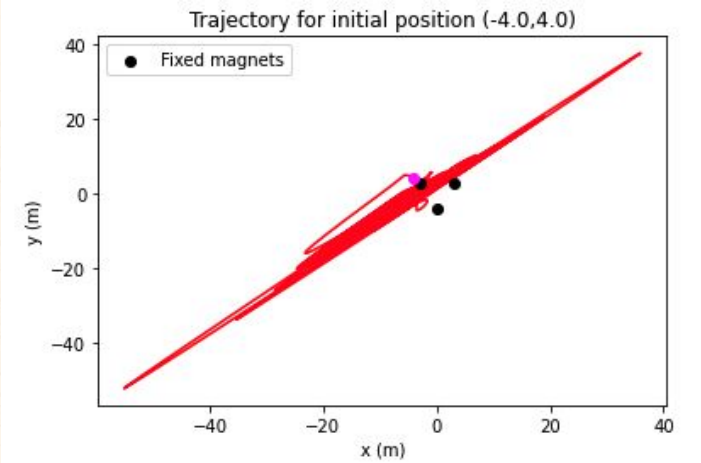


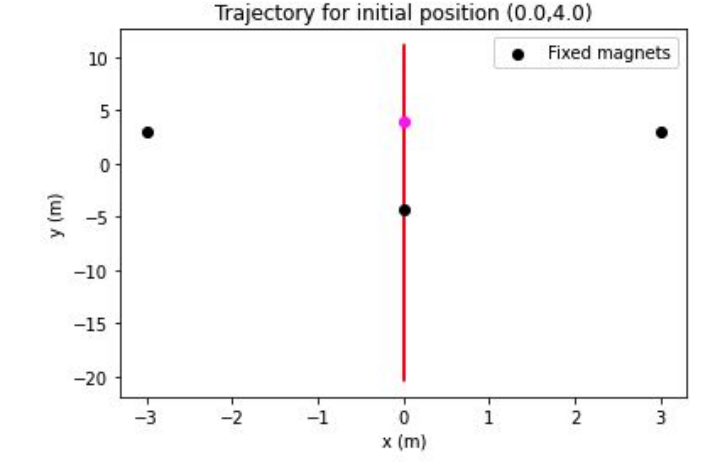




Basins of attraction







THANK YOU!

References:

https://iptnet.info/

https://compute.dawsoncollege.qc.ca/winter-2020/modelling-physical-systems-odes/magnetic-pendulum-trajectory/

https://www.math.hmc.edu/~dyong/math164/2006/win/finalreport.pdf

https://softologyblog.wordpress.com/2012/04/11/chaotic-oscillating-magnetic-pendulum-simulation/