

Série de Taylor - Ciclóide

Nome Completo – Turma ??? (Matricula:)

DD/MM/AAAA

Resumo

O experimento consiste em aproximar o movimento de uma cicloide utilizando a série de Taylor.

Materiais e ferramentas

Celular ou outro equipamento com câmera;

Pneu (fornecido pelo Laboratório)

Media Player Classic (ou outro player que forneça a precisão do tempo do vídeo em milissegundos, como o avidemux).

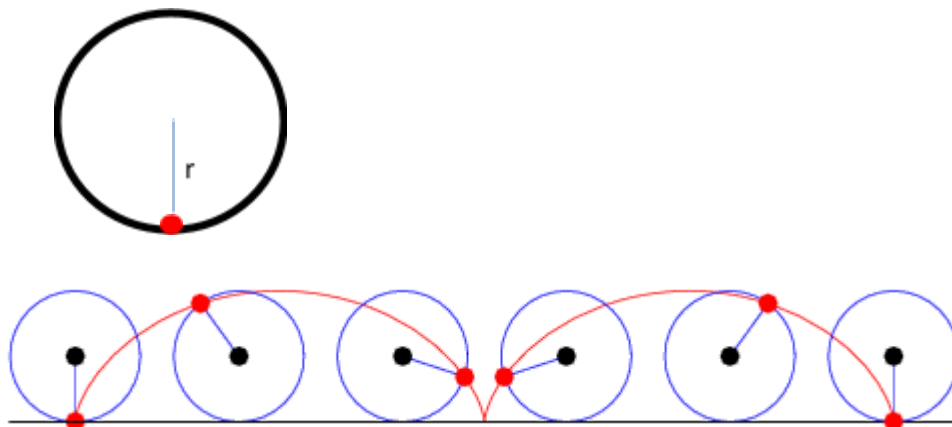
Paint (ou outro editor de imagem que forneça a posição de determinado pixel na imagem)

Metodologia

No presente trabalho, o grupo irá filmar o movimento realizado por um ponto sobre o círculo rolando, formando uma cicloide.

“Chama-se **ciclóide** a curva definida por um ponto de uma circunferência que rola sem deslizar sobre uma reta.”

Fonte: Wikipedia.



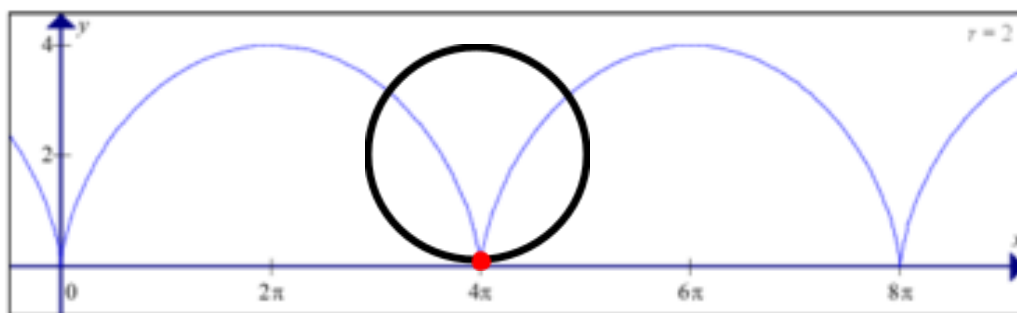


Figura 1- Desenho de uma cicloide gerada por uma circunferência de raio $r=2$

O aluno deve posicionar a câmera em cima de uma mesa, e filmar o movimento do pneu sobre a mesa e à frente da câmera de forma que este faça aproximadamente duas voltas, como mostrado na Figura 1. A câmera deve estar nivelada e permanecer parada durante a filmagem (em modo paisagem).

O filme deve ser enviado para o computador e, a partir deste, os dados para o experimento serão coletados.

Após o arquivo estar disponível no computador, abra o arquivo utilizando o Media Player Classic (MPC), ou outro player.

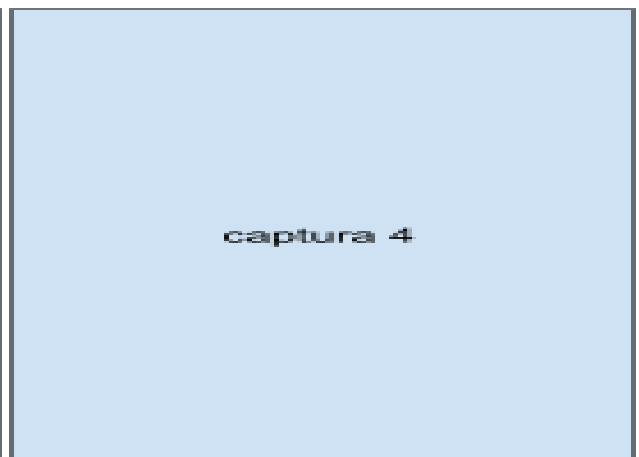
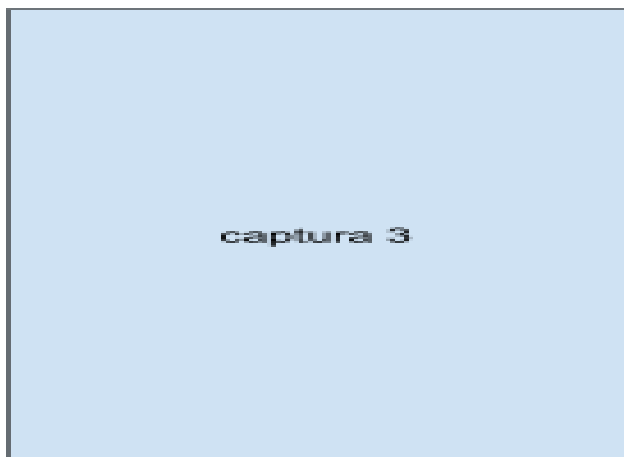
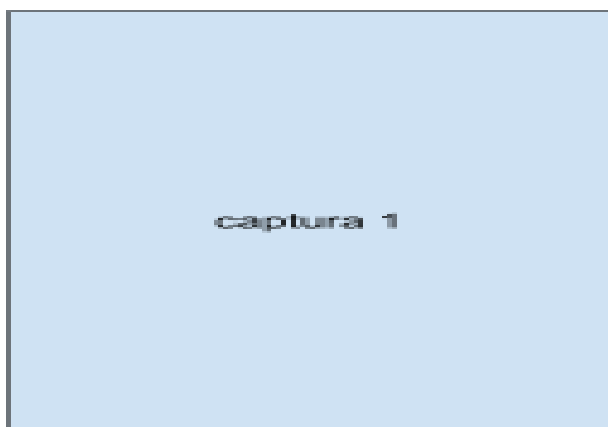
Experimento

A partir do vídeo, capture 5 imagens utilizando o software Media Player Classic (MPC):

- 1- Abra o vídeo com o Media Player Classic (MPC);
- 2- Selecione os quadros desejados com Ctrl+ “seta direita” ou Ctrl+ “seta esquerda”;
- 3- Na aba “arquivo” clique em “salvar imagem” (ou “alt+i”);

As imagens capturadas devem ser as seguintes:

- 1 - Quando a marca toca o solo (Momento inicial);
- 2 - A marca atinge a altura máxima do movimento pela primeira vez;
- 3 - A marca toca novamente o solo
- 4 - A marca atinge a altura máxima pela segunda vez;
- 5 - A marca toca novamente o solo.





Para cada imagem capturada, deve-se descobrir a posição, em pixels, do ponto marcado no pneu. Para tal, utilize o Paint.

1- Abra cada imagem utilizando o “Paint”.

2- Coloque o mouse exatamente em cima de onde deseja saber a posição em pixels.

3- A posição do mouse sobre a imagem, em pixels, aparece na extremidade inferior esquerda da janela do Paint. Anote as coordenadas x e y para cada imagem capturada.

	1	2	3	4	5
x					
y					

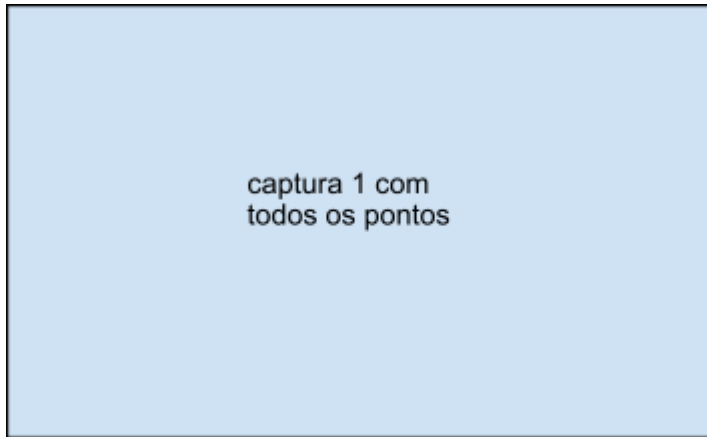
Abra a primeira imagem capturada do vídeo utilizando o Colaboratory e plote sobre essa imagem todos os pontos capturados.

```
#para carregar as figuras a partir do colab. Só rodar 1 vez!
from google.colab import files
from io import BytesIO
from PIL import Image
uploaded = files.upload()
img = Image.open(BytesIO(uploaded['c1.jpg'])) #o nome da figura deve estar igual
ao da figura a ser carregada

#Importar módulos
import matplotlib.pyplot as plt
import numpy as np

#carregar e mostrar a imagem original
img = plt.imread('c1.jpg') #o nome deve ser igual ao carregado.
plt.imshow(img)
```

```
#plotar sobre a imagem os pontos capturados
x= np.array([x1,x2,x3,...])#mudar os valores de x e y
y= np.array([y1,y2,y3,...])
plt.plot(x,y,'ro')
plt.show()
```



OBS: Caso o eixo “y” pareça invertido, deve-se inverter os valores de y. Para tanto, descubra a dimensão y da imagem e subtraia dessa dimensão, todos os valores de y encontrados. Refaça o gráfico anterior caso isso tenha acontecido.

O movimento da cicloide no plano [x,y] pode ser modelado com as seguintes equações:

$$x = r(\theta - \sin(\theta))$$

$$y = r(1 - \cos(\theta))$$

em que r é o raio do pneu, e θ é o ângulo de rotação do pneu (quando $\theta = 2\pi$, por exemplo, o pneu terá dado uma volta completa).

Utilizando duas imagens coletadas em que o ponto está encostando na mesa, e sabendo que a distância entre estes dois pontos será igual a circunferência do pneu, descubra qual o raio, em pixel, do pneu.

R= _____

Com tais dados, o aluno deve criar duas funções que aproximam o movimento da cicloide (uma para x e outra para y, ambas em função de θ) por uma série de Taylor com n termos em torno de θ_0 (em que n e θ_0 devem ser argumentos da função, assim como θ).

O algoritmo abaixo encontra Taylor da função que implementa o movimento em x.

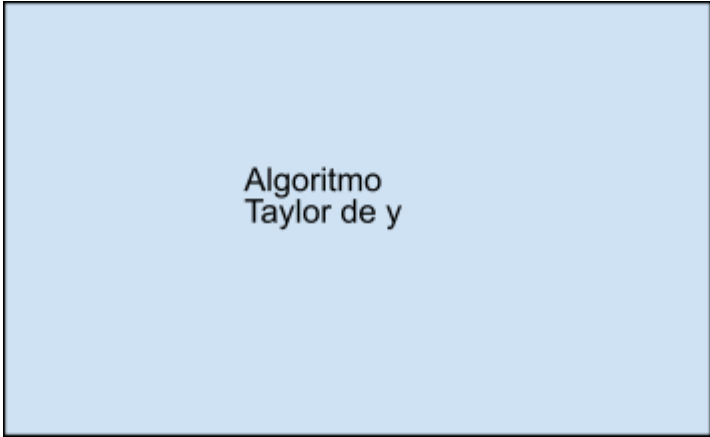
```
def taylorx(r,t0,t,n):
    h = t - t0
    #dois primeiros termos
    tay = r*(t0 - np.sin(t0)) + (r - r*np.cos(t0))*h
```

```

#outros termos
for i in range(2,n+1):
    if i%4 == 1:
        d = - r*np.cos(t0)
    if i%4 == 2:
        d = r*np.sin(t0)
    if i%4 == 3:
        d = r*np.cos(t0)
    if i%4 == 0:
        d = - r*np.sin(t0)
    tay = tay + (d*h**i)/np.math.factorial(i)
return tay

```

Modifique o algoritmo para encontrar Taylor da função que implementa o movimento em y.



Algoritmo
Taylor de y

Chame suas funções de taylor para $\theta_0=0$ e vários ângulos θ (variando de zero a 4π). Plote o resultado de x por y para ver a cicloide aproximada por Taylor, como na Figura 1. Varie o número de termos da série de Taylor (n) para ver quantos termos são necessários para aproximar o movimento completo com boa precisão (poucos termos aproximam bem o início do movimento, mas para aproximar por completo normalmente temos que utilizar entre 30 e 40 termos).

```

import numpy as np
import matplotlib.pyplot as plt

xc= np.array([x1,x2,x3,...])
yc= np.array([y1,y2,y3,...])
r= ??? #Raio do pneu em pixels

print(r)

```

```

#cicloide real
t = np.linspace(0,np.pi*4, 100)
xcr = r*(t - np.sin(t))
ycr = r*(1 - np.cos(t))
fig, ax = plt.subplots(figsize=(15, 15))
plt.grid()
plt.plot(xcr,ycr,'m-.')

#chama taylor para cada t e cria x e y
t0 = 0
n = 40 #ordem taylor
x=np.zeros(len(t))
y=np.zeros(len(t))

for i in range(len(t)):
    x[i]=taylorx(r,t0,t[i],n)
    y[i]=taylory(r,t0,t[i],n)
plt.plot(x,y)

```

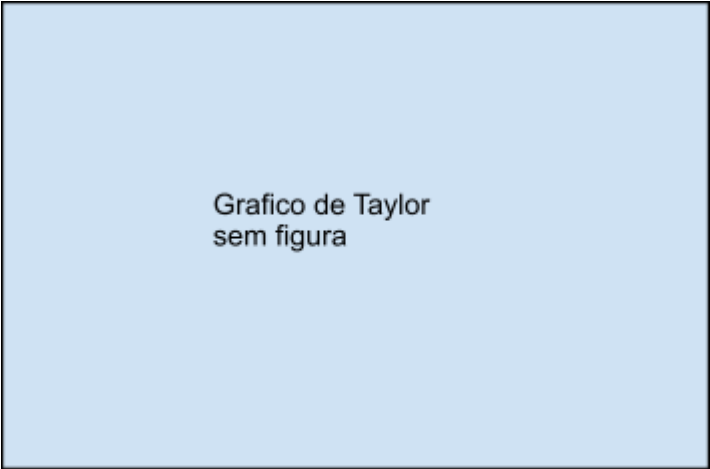


Grafico de Taylor
sem figura

Para visualizar melhor os resultados, plote sobre a primeira imagem capturada todos os pontos e o resultado. Altere o resultado de Taylor para corresponder ao início do movimento.

Grafico de
Taylor com
captura 1 e
pontos