5-Noções Básicas de VPython

April 20, 2021

1 Informes iniciais

1. A interface do VPython, juntamente com as suas figuras, aparece na mesma célula de importação do vpython pela primeira vez após o kernel ser reiniciado. Então, após executar a célula para se gerar as figuras, **o usuário irá vizualizá-las na célula de importação da biblioteca.** Nesse caso, logo abaixo de **3. Importando a Biblioteca VPython**;
2. Caso o usuário queira "limpar" as figuras na interface, é preciso reiniciar o kernel e as saídas, indo no menu em `Kernel -> Restart & Clear Output`, se estiver no Jupyter Notebook;
3. É preciso lembrar de **sempre** executar a célula para importar a biblioteca VPython após reiniciar o kernel, uma vez que ela é a responsável por proporcionar a criação das figuras e afins.

2 VPython

VPython é uma biblioteca gráfica do python capaz de proporcionar a inserção de figuras geométricas tridimensionais – elipsoides, paralelepípedos, pirâmides, trapézios, entre outros – em sua interface, permitindo que essas possam ser manipuladas de acordo com mudanças em seus parâmetros. Assim, a alteração em seus valores pode modelar algumas situações que envolvem o movimento, uma vez que eles estão associados à posição, massa, tamanho e, até mesmo, velocidade. Nesse sentido, aplicando determinadas equações na construção dos seus algoritmos, é possível criar simuladores para tipos específicos de movimento, como os amplamente abordados na disciplina de Física.

3 Importando a Bibioteca VPython

[ ]:
```
from vpython import * #SEMPRE antes de desenvolver o código
```

4 Gerando as Figuras

Na construção dos simuladores, é preciso se atentar à interface do vpython em que as figuras são inseridas: um espaço cartesiano tridimensional com origem em seu centro. Para facilitar a inserção desses objetos na interface, muitos parâmetros relacionados a eles recebem valores do tipo vetorial nas três coordenadas espaciais (`vec(x, y, z)`), a exemplo da posição na interface e as dimensões do objeto. Outrossim, apesar de não serem necessários para uma criação inicial desses objetos, outros parâmetros, como cor, orientação do seu eixo e opacidade, são interessantes para ilustrar melhor algumas simulações, sendo, em alguns casos, essenciais para uma melhor visualização dessas. Por fim, para possibilitar a alteração dos seus valores, é interessante associar cada objeto criado a uma variável capaz de orientá-lo.

Para mais informações sobre a variável vetorial, ver: https://www.glowscript.org/docs/VPythonDocs/vector.html

4.1 Paralelepípedos

Para a criação do paralelepípedo, utilizamos uma variável do tipo `box`, associando parâmetros referentes a sua posição (`pos=vec(x, y, z)`) e tamanho (`size=vec(x, y, z)`).

[ ]:
```
ratangulo_a = box(pos=vec(-2,2,0), size=vec(2, 3, 4))
```

Além desses parâmetros, também é possível se adicionar outros parâmetros para uma melhor vizualização, como cor (`color`) e opacidade(`opacity`), orientação/sentido (`axis = vec(x, y, z)`)

[ ]:
```
ratangulo_b = box(pos=vec(3,2,0), size=vec(2, 3, 4), color = color.red,␣
↪opacity=1, axis = vec(1, 0.5, -1))
ratangulo_c = box(pos=vec(-2,-2,0), size=vec(1, 1, 1), color = color.yellow,␣
↪opacity=0.1)
```

Para mais informações sobre o paralelepípedo, ver: https://www.glowscript.org/docs/VPythonDocs/box.html

Para mais informações sobre cores e opacidade, ver: https://www.glowscript.org/docs/VPythonDocs/color.html

Para mais informações sobre orientação/sentido, ver: https://www.glowscript.org/docs/VPythonDocs/cylinder.htı

Reinicie o kernel e limpe as saídas (ver Informes Iniciais) e comece a execução dos códigos a partir da célula abaixo para uma melhor facilidade na vizualização

4.2 Esferas

Nesse caso, utilizamos uma variável do tipo `sphere`, associando parâmetros referentes a sua posição (`pos=vec(x, y, z)`) e raio (`radius`). Além disso, os parâmetros são os mesmos dos retângulos, exceto que, devido à sua forma, a sua inclinação/sentido não é necessário.

[ ]:
```
esfera = sphere(pos = vec(0, 0, 0), radius = 2, color = color.red, opacity=0.1)
```

Para mais informações sobre retângulos, ver: https://www.glowscript.org/docs/VPythonDocs/sphere.html

Reinicie o kernel e limpe as saídas (ver Informes Iniciais) e comece a execução dos códigos a partir da célula abaixo para uma melhor facilidade na vizualização

4.3 Cilindros

Nesse caso, utilizamos uma variável do tipo `cylinder`, associando parâmetros referentes a sua posição (`pos=vec(x, y, z)`), raio (`radius`) e comprimento/altura (sendo definida de acordo com os valores atribuidos em sua orientação/sentido (`axis=vec(x, y, z)`).

Além disso, os parâmetros são os mesmos dos retângulos.

[ ]:
```
cilindro = cylinder(pos=vector(-2,0,0), axis=vector(4,0,0), radius=0.8, color =␣
↪color.yellow, opacity=1)
```

Para mais informações sobre cinlíndros, ver: https://www.glowscript.org/docs/VPythonDocs/cylinder.html

Reinicie o kernel e limpe as saídas (ver Informes Iniciais) e comece a execução dos códigos a partir da célula abaixo para uma melhor facilidade na vizualização

4.4 Textos

Para inserir textos, utilizamos variáveis do tipo `label`, tendo como parâmetros a posição(`pos(vec(x, y, z))`, texto (`text='texto'`), comprimento (`length=x`), altura (`height=y`), espaçamento entre as letras(`space`), espessura da borda (`border`) e tipo de fonte (`font`).

[ ]:
```
texto = label(pos=vec(0, 0, 0), text= 'texto', space=30, height=16, border=4,␣
↪font='sans')
```

Para masi informações sobre os parâmetros dos textos, ver: https://www.glowscript.org/docs/VPythonDocs/text.html

Reinicie o kernel e limpe as saídas (ver Informes Iniciais) e comece a execução dos códigos a partir da célula abaixo para uma melhor facilidade na vizualização

4.5 Outras Figuras

Para além das figuras apresentadas, o VPython possibilita a criação de diversas outras, tendo parâmetros iguais aos apresentados, além de outros específicos de cada forma. Abaixo, segue uma pequena amostragem:

[ ]:
```
cubo = box(pos=vector(-1,0,0), color=color.purple, opacity=1)
paralelepipedo = box(pos=vector(1,-1,0), color=color.green, size=vector(2,1,1))
seta = arrow(pos=vector(-1,-1.3,0), color=color.cyan)
Cone = cone(pos=vector(2,0,0), axis=vector(0,1,-.3), color=color.
↪blue,size=vector(2,1,1))
elipsoide = sphere(pos=vector(-1.5,1.5,0), color=color.white, size=.
↪4*vector(3,2,1))
anel = ring(pos=vector(-0.6,-1.3,0), size=vector(0.2,1,1), color=color.red)
helice = helix(pos=vector(-3,-1.3,0), radius=0.2, constant=1,thickness=0.
↪1,coils=5,color= color.yellow)
```

Para mais informações sobre as figuras, ver: https://www.glowscript.org/docs/VPythonDocs/shapes.html