Rotacion Escalado y traslacion

October 9, 2024

0.1 UNIVERSIDAD AUTONOMA DEL ESTADO DE MEXICO

1 INGENIERIA EN COMPUTACION

RAUL ALEJANDRO CALDERON HERNANDEZ

GRAFICACION COMPUTACIONAL

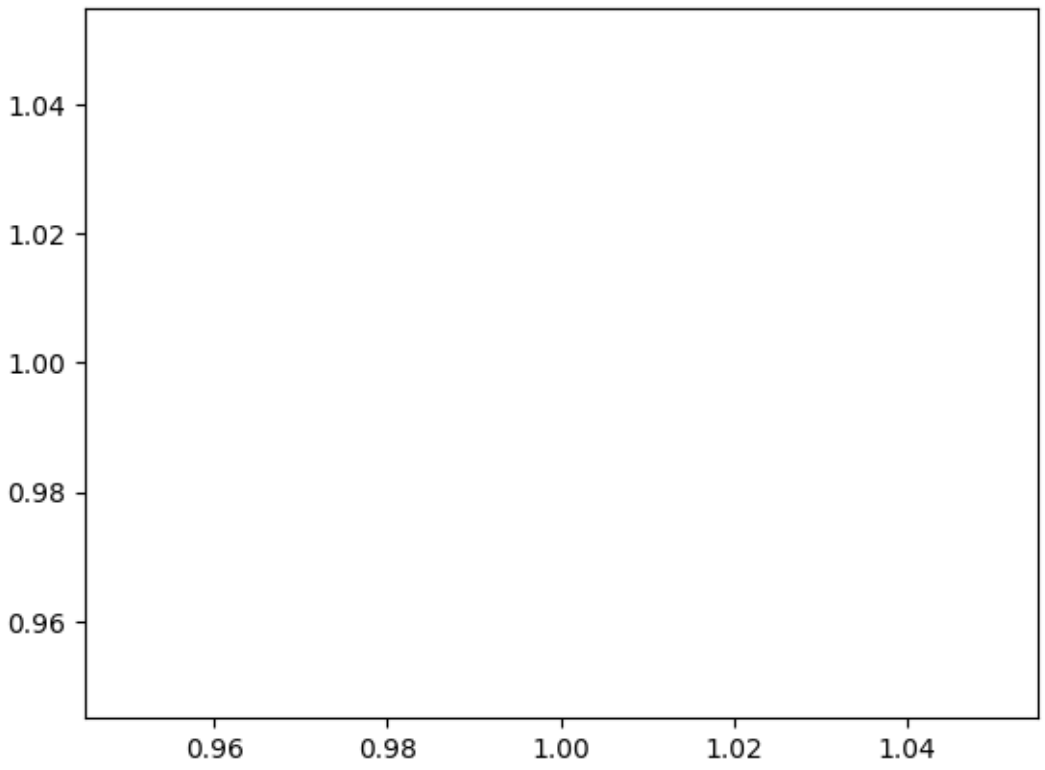
TRANSFORMACIONES GEOMETRICAS

[4]:
```
import matplotlib.pyplot as plt
```

Importamos matplotlib para graficar

[5]:
```
plt.plot([1],[1])
```

[5]:
```
[<matplotlib.lines.Line2D at 0x22ea6704450>]
```



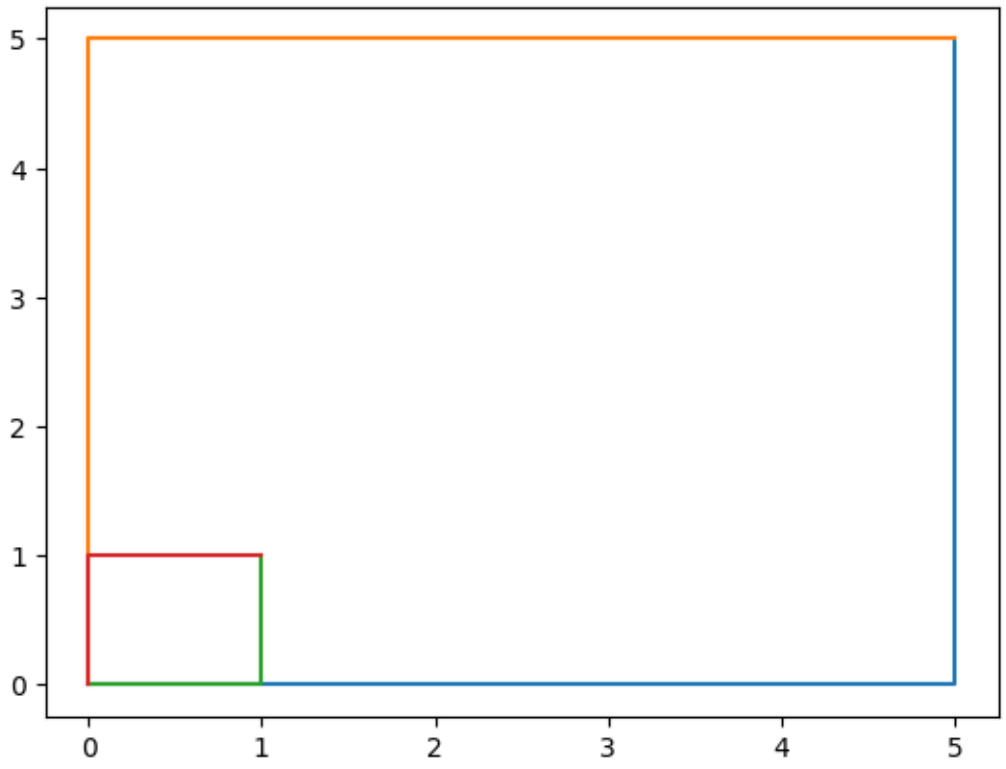
Construimos el plano de 1 por 1 para que nos de una demostracion de como quedaria el plano

1.1 ESCALAR

es una operación en geometría que cambia el tamaño de un objeto sin alterar su forma. Se realiza multiplicando las coordenadas de los puntos de un objeto por un valor constante, llamado factor de escala. Y si es menor a 1 se hace pequeña a la imagen original, pero si es mayor que uno entonces se hace mas grande

[24]:
```
numero = 5
plt.
  ↪plot([0*numero,1*numero,1*numero],[0*numero,0*numero,1*numero],[0*numero,0*numero,1*numero]
plt.plot([0,1,1],[0,0,1],[0,0,1],[0,1,1])
```

[24]:
```
[<matplotlib.lines.Line2D at 0x22eab9a28d0>,
 <matplotlib.lines.Line2D at 0x22eab9b8410>]
```

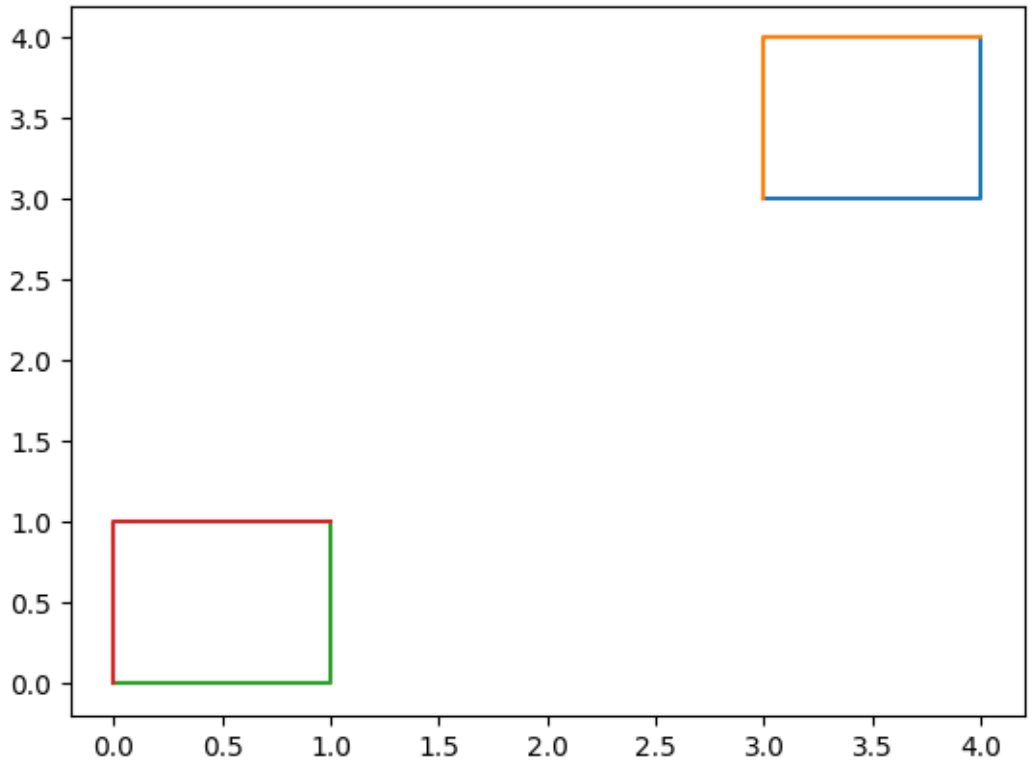


1.2 TRASLACION

consiste en mover todos los puntos de una figura una cierta distancia en una dirección determinada, sin cambiar su forma, tamaño o orientación.

[25]:
```
numero = 3
plt.
  ↪plot([0+numero,1+numero,1+numero],[0+numero,0+numero,1+numero],[0+numero,0+numero,1+numero]
plt.plot([0,1,1],[0,0,1],[0,0,1],[0,1,1])
```

[25]:
```
[<matplotlib.lines.Line2D at 0x22eab987690>,
 <matplotlib.lines.Line2D at 0x22eab9614d0>]
```



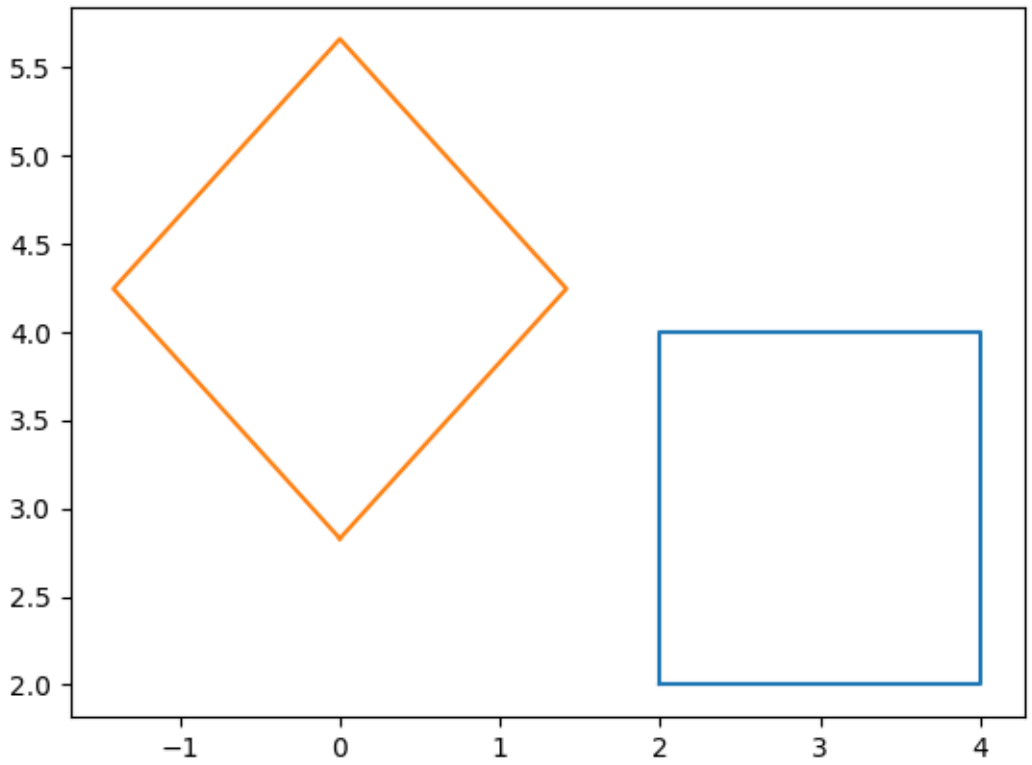
1.3 ROTACION

gira una figura alrededor de un punto fijo (generalmente el origen, pero puede ser cualquier otro punto). La rotación en 2D se especifica por un ángulo y el sentido de giro (sentido horario o antihorario).

[26]:
```
import numpy as np
x = [2,2,4,4,2]
y = [2,4,4,2,2]
plt.plot(x,y)
angulo = 45
t = np.radians(angulo)
rotarx = np.cos(t) * np.array(x) - np.sin(t) * np.array(y)
rotary = np.sin(t) * np.array(x) + np.sin(t) * np.array(y)
```

```
plt.plot(rotarx,rotary)
```

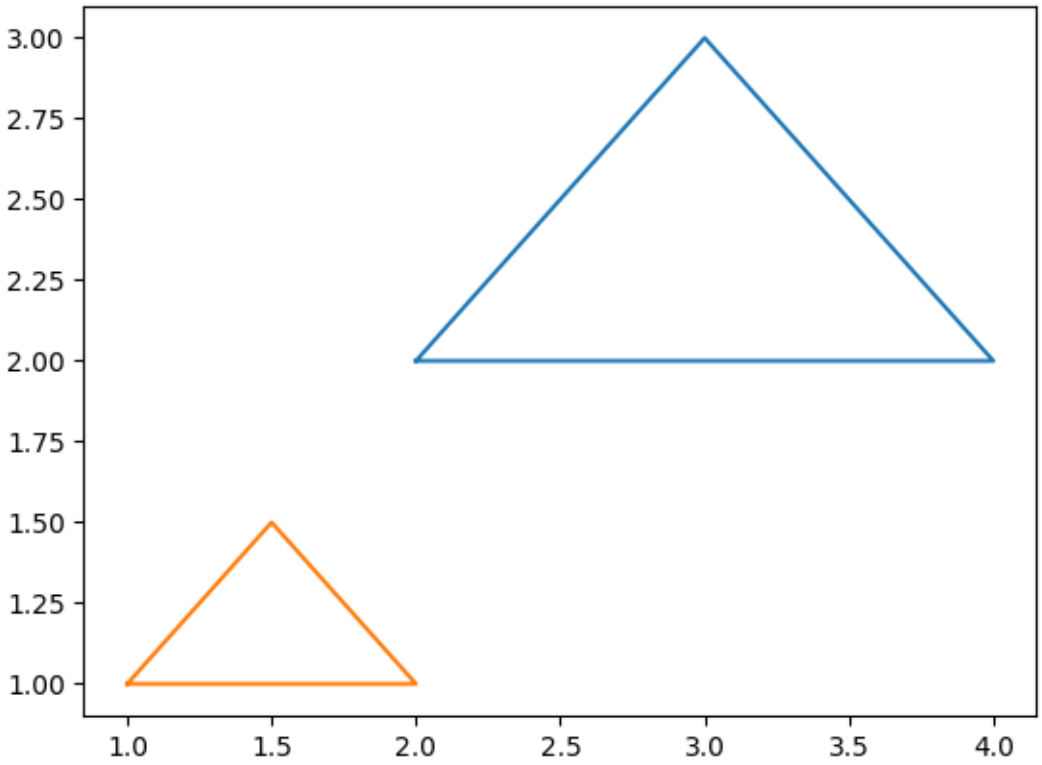
[26]: [<matplotlib.lines.Line2D at 0x22eab9dea10>]



1.4 TRIANGULO

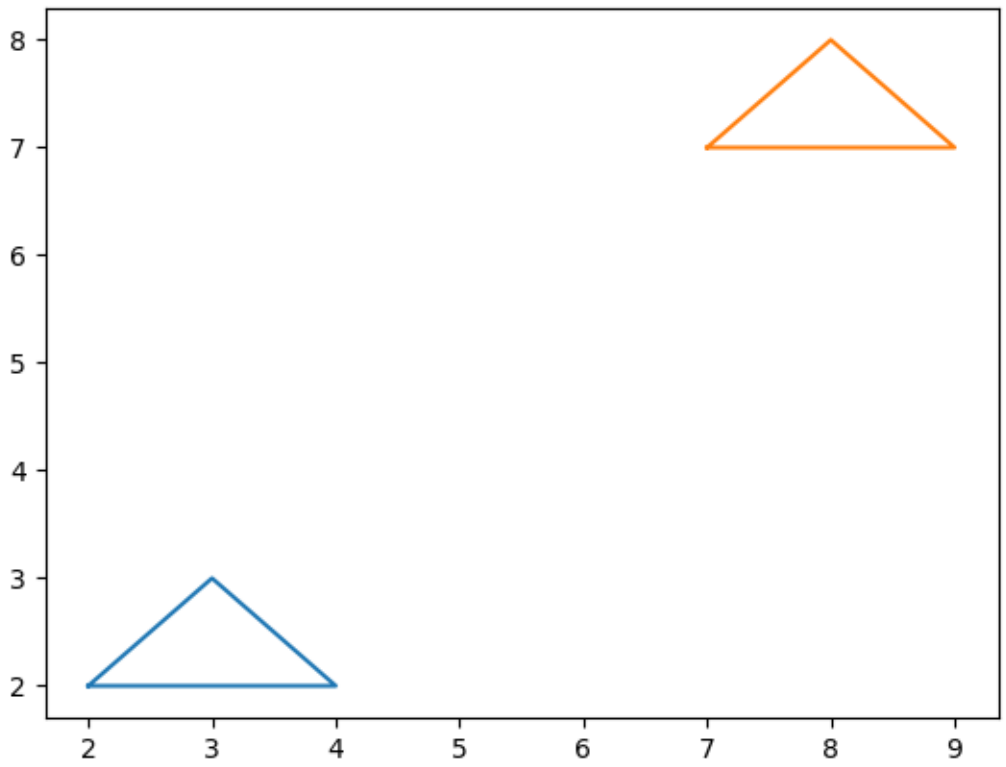
[27]:
```
numero = 0.5
x = [2,3,4,2]
y = [2,3,2,2]
plt.plot(x,y)
x = [2*numero,3*numero,4*numero,2*numero]
y = [2*numero,3*numero,2*numero,2*numero]
plt.plot (x,y)
```

[27]: [<matplotlib.lines.Line2D at 0x22eab8759d0>]



[28]:
```
numero = 5
x = [2,3,4,2]
y = [2,3,2,2]
plt.plot(x,y)
x = [2+numero,3+numero,4+numero,2+numero]
y = [2+numero,3+numero,2+numero,2+numero]
plt.plot (x,y)
```

[28]: [<matplotlib.lines.Line2D at 0x22eabb74310>]



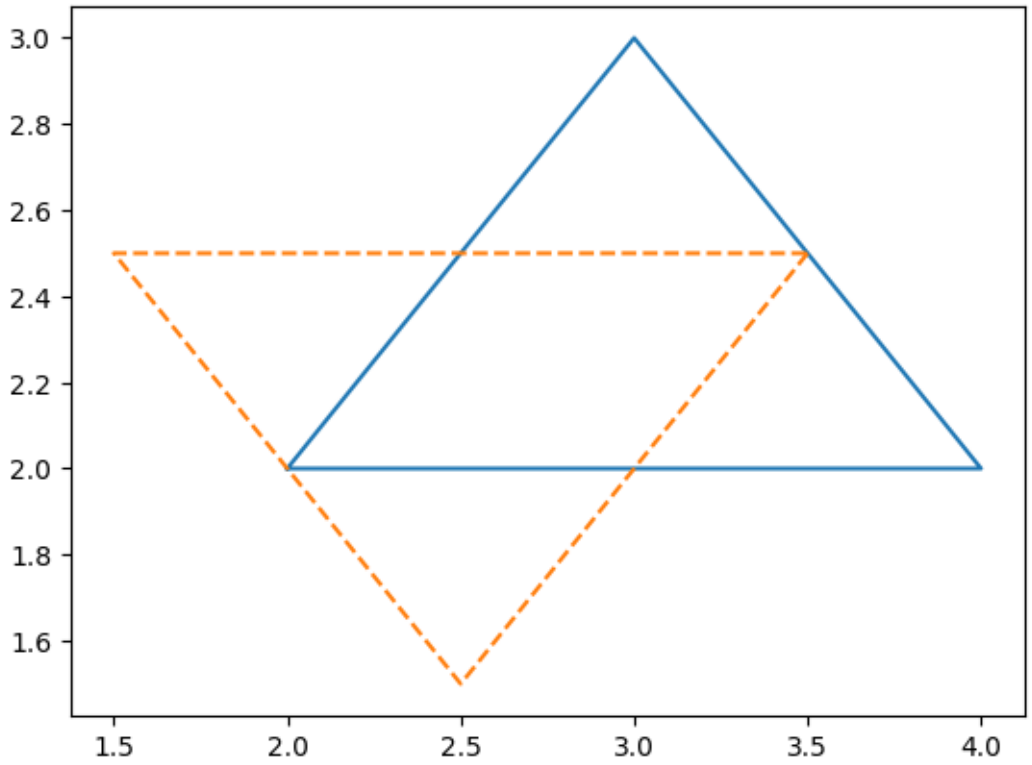
[29]:
```
import numpy as np
x = [2,3,4,2]
y = [2,3,2,2]
plt.plot(x,y)
angulo = 180
t = np.radians(angulo)
xc = np.mean(x)  # Calcular el centro de la figura en x
yc = np.mean(y)  # Calcular el centro de la figura en y
x_centrado = np.array(x) - xc
y_centrado = np.array(y) - yc

rotarx = np.cos(t) * x_centrado - np.sin(t) * y_centrado
rotary = np.sin(t) * x_centrado + np.cos(t) * y_centrado

rotarx += xc
rotary += yc

plt.plot(rotarx, rotary,linestyle="--")
```

[29]: [<matplotlib.lines.Line2D at 0x22eabb8ff10>]



Como observamos estas transformaciones afectan directamente a nuestra figura, ya sea como una traslacion rotacion o movimiento que la figura pueda tener, tambien podemos combinarlas para tener mas figuras, esto con mayor lineas de codigo, pero asi se hace un pequeño programa facil de hacer para las transformaciones geometricas clasicas y basicas

Podemos intuir que las transformaciones tambien son una parte fundamental dentro de las geometria, y en si de la graficacion,

[ ]: