Traslacion,_rotacion_y_escalado

October 9, 2024

1 UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MÉXICO

2 CU UAEM ZUMPANGO

Ingenieria en computación

Graficación Computacional

Alumno: Diego Gómez Tagle González Diego

Docente: Hazem Álvarez Rodriguez

9 de octubre del 2024

[1]:
```
import matplotlib.pyplot as plt
import numpy as np
```

Muestra las siguientes figuras:

- Traslación
- Rotación
- Escalamiento

2.1 Traslación

La traslación es una transformación que desplaza una figura geométrica de una posición a otra sin cambiar su forma, tamaño u orientación. Consiste en sumar un valor constante a las coordenadas de todos los puntos de la figura.

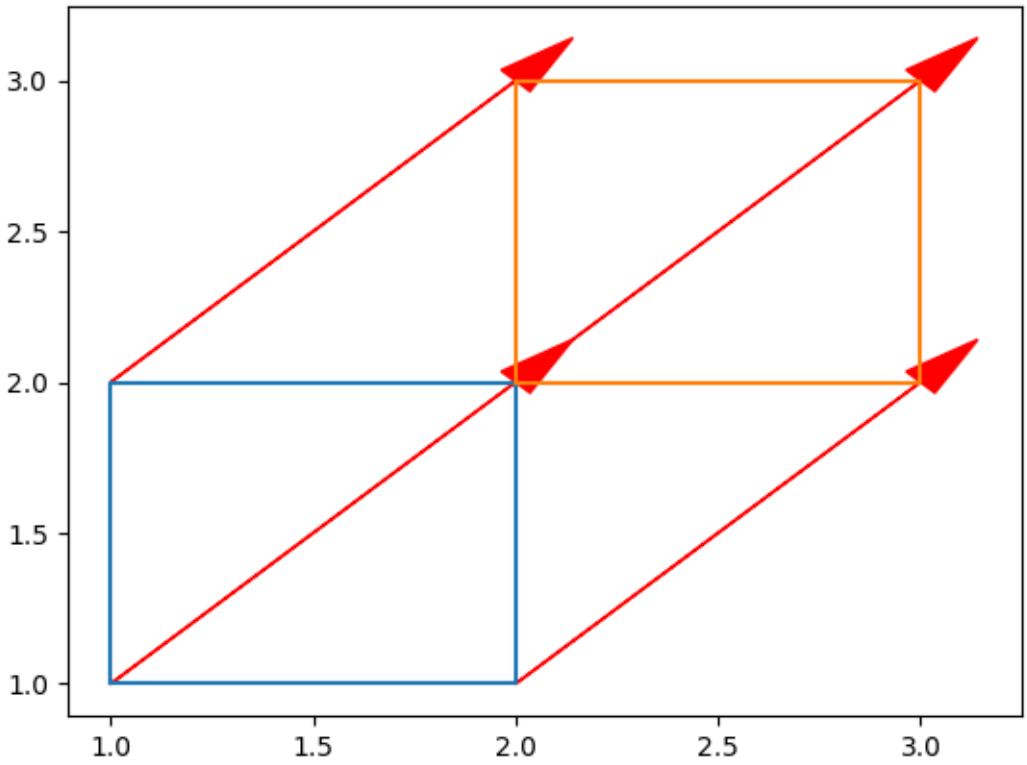
Fórmulas para la traslación: x =x+tx

y =y+ty

Donde y son los desplazamientos en los ejes y.

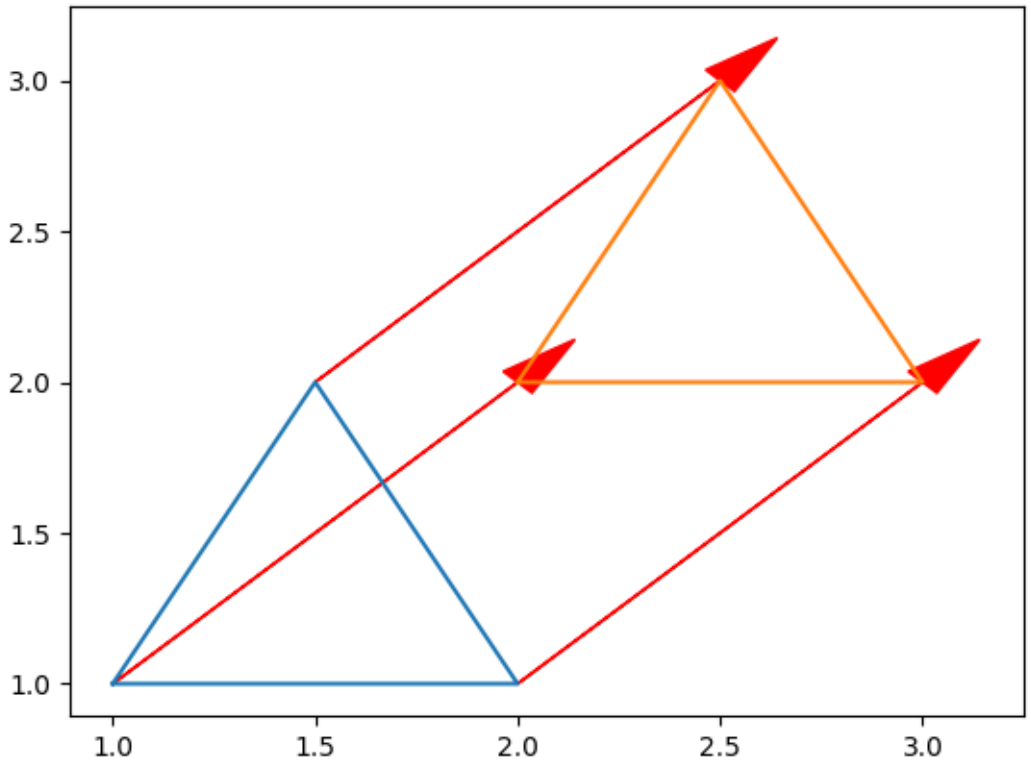
[2]:
```
plt.plot([1,2,2,1,1], [1,1,2,2,1])
plt.plot([2,3,3,2,2], [2,2,3,3,2])
plt.arrow (2, 2, 1, 1, head_width=0.1, head_length=0.2, fc='r', ec='r')
plt.arrow (1, 2, 1, 1, head_width=0.1, head_length=0.2, fc='r', ec='r')
plt.arrow (1, 1, 1, 1, head_width=0.1, head_length=0.2, fc='r', ec='r')
plt.arrow (2, 1, 1, 1, head_width=0.1, head_length=0.2, fc='r', ec='r')
```

[2]:
```
<matplotlib.patches.FancyArrow at 0x794a3c6eae60>
```



[3]:
```
plt.plot([1,2,1.5,1], [1,1,2,1])
plt.plot([2,3,2.5,2], [2,2,3,2])

plt.arrow(2, 1, 1, 1, head_width=0.1, head_length=0.2, fc='r', ec='r')
plt.arrow(1.5, 2, 1, 1, head_width=0.1, head_length=0.2, fc='r', ec='r')
plt.arrow(1, 1, 1, 1, head_width=0.1, head_length=0.2, fc='r', ec='r')
```

[3]:
```
<matplotlib.patches.FancyArrow at 0x794a0bd47370>
```



2.2 Rotación

La rotación es una transformación que gira una figura geométrica alrededor de un punto fijo (típicamente el origen) por un ángulo determinado. La forma y el tamaño de la figura no cambian, solo su orientación.

Fórmulas para la rotación en un ángulo alrededor del origen:

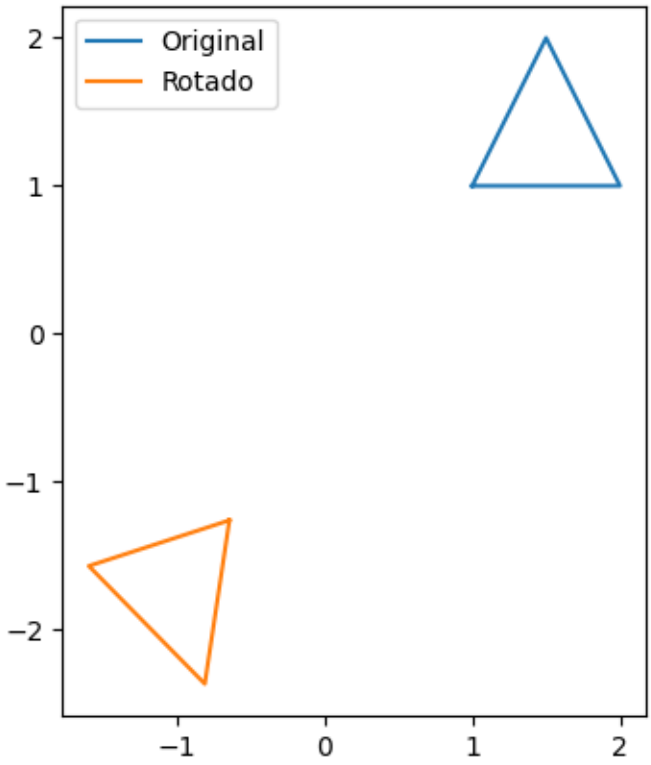
$$x = x\cos(\theta) - y\sin(\theta)$$

$$y = x\sin(\theta) + y\cos(\theta)$$

Donde es el ángulo de rotación en radianes. Un ángulo positivo indica una rotación en sentido antihorario y un ángulo negativo indica una rotación en sentido horario.

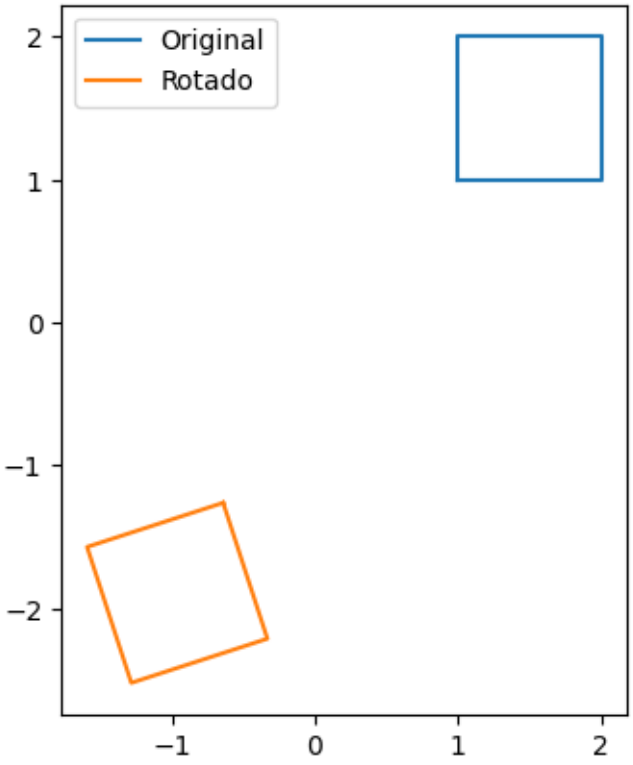
[5]:
```
import numpy as np
import matplotlib.pyplot as plt

# Coordenadas del triángulo
x = [1, 2, 1.5, 1]
y = [1, 1, 2, 1]

# Grado de rotación
theta = 198
```

```
# Función para rotar la figura
def rotar(x, y, theta):
    # Convertir listas a arrays de numpy
    x = np.array(x)
    y = np.array(y)

    # Convertir grados a radianes
    theta_rad = np.radians(theta)

    # Aplicar las fórmulas de rotación
    xr = x * np.cos(theta_rad) - y * np.sin(theta_rad)
    yr = x * np.sin(theta_rad) + y * np.cos(theta_rad)

    return xr, yr

# Rotar el triángulo
xx, yy = rotar(x, y, theta)

# Graficar el triángulo original y rotado
plt.plot(x, y, label="Original")
plt.plot(xx, yy, label="Rotado")

plt.legend()
plt.gca().set_aspect('equal', adjustable='box')
plt.show()
```



[6]:
```python
# Coordenadas del cuadrado
x = [1, 2, 2, 1, 1]  # Agregué el primer punto al final para cerrar la figura
y = [1, 1, 2, 2, 1]

# Grado de rotación
theta = 198

# Función para rotar la figura
def rotar(x, y, theta):
    # Convertir listas a arrays de numpy
    x = np.array(x)
    y = np.array(y)

    # Convertir grados a radianes
    theta_rad = np.radians(theta)

    # Aplicar las fórmulas de rotación
    xr = x * np.cos(theta_rad) - y * np.sin(theta_rad)
    yr = x * np.sin(theta_rad) + y * np.cos(theta_rad)

    return xr, yr
```

```
# Rotar el cuadrado
xx, yy = rotar(x, y, theta)

# Graficar el cuadrado original y rotado
plt.plot(x, y, label="Original")
plt.plot(xx, yy, label="Rotado")

plt.legend()
plt.gca().set_aspect('equal', adjustable='box')
plt.show()
```



2.3 Escalamiento

El escalamiento es una transformación que cambia el tamaño de una figura geométrica sin alterar su forma. Esta transformación puede ser uniforme (cuando las dimensiones se modifican en la misma proporción en todas las direcciones) o no uniforme (cuando se cambia el tamaño de manera diferente en cada dirección)

Fórmulas para el escalamiento:

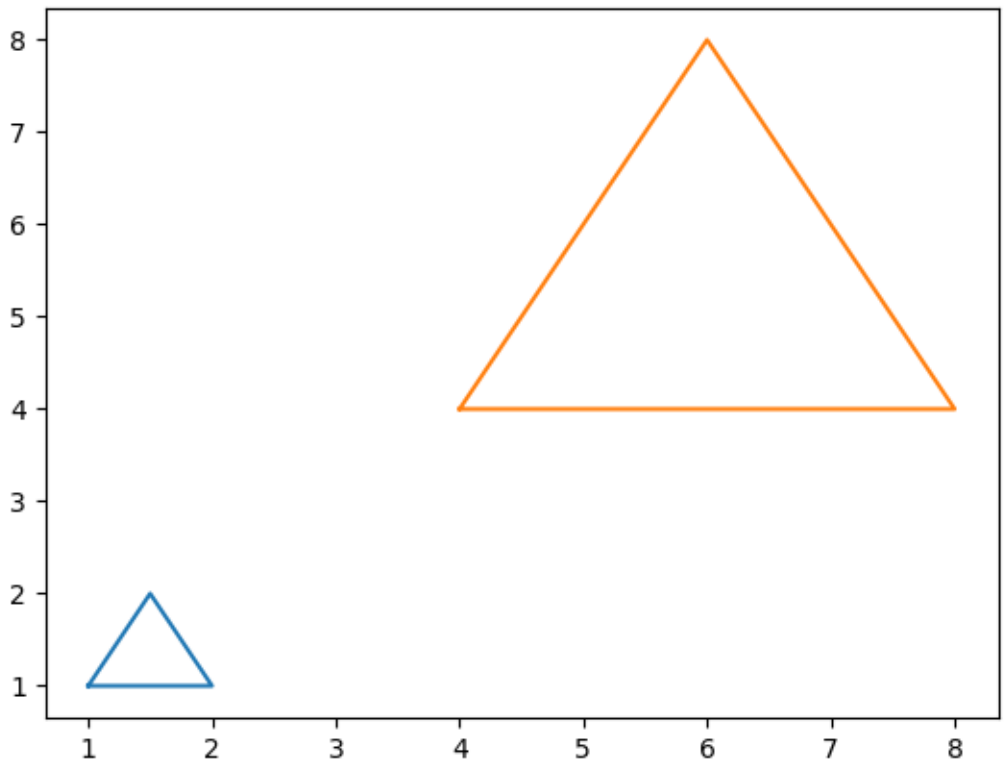
=

= y

Donde y son los factores de escala en los ejes

[11]:
```
#coordenadas de la figura origial

c = [1, 2, 1.5, 1]
r = [1, 1, 2, 1]

#el escalamiento
Sx, Sy = 4, 4

#formula

cc =  [Sx*x for x in c]
rr = [Sy*y for y in r]

#muestra original y escalonada

plt.plot(c,r)
plt.plot(cc,rr)
```

[11]:
```
[<matplotlib.lines.Line2D at 0x794a0be21b10>]
```



[12]:
```
c=[1,2,2,1,1]
r=[1,1,2,2,1]
#el escalamiento
Sx, Sy = 4, 4

#formula

cc =  [Sx*x for x in c]
rr = [Sy*y for y in r]

#muestra original y escalonada

plt.plot(c,r)
plt.plot(cc,rr)
```

[12]:
```
[<matplotlib.lines.Line2D at 0x794a0be2ec50>]
```