```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Load the image
image = cv2.imread(r"C:\Users\admin\Desktop\abdul kalam.jpg")

# Helper function to display images
def display_image(title, img):
    plt.figure(figsize=(6, 6))
    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    plt.title(title)
    plt.axis('off')
    plt.show()

# 1. Translation
tx, ty = 100, 50  # Translate x by 100 pixels, y by 50 pixels
M_translation = np.float32([[1, 0, tx], [0, 1, ty]])
translated_image = cv2.warpAffine(image, M_translation, (image.shape[1], image.shape[0]))
display_image("Translated Image", translated_image)

# 2. Rigid Transformation (Rotation + Translation)
angle = 30  # Rotate by 30 degrees
center = (image.shape[1] // 2, image.shape[0] // 2)
M_rigid = cv2.getRotationMatrix2D(center, angle, 1.0)
```

```
rigid_transformed_image = cv2.warpAffine(image, M_rigid, (image.shape[1], image.shape[0]))

display_image("Rigid Transformation (Rotation + Translation)", rigid_transformed_image)


# 3. Similarity Transformation (Rotation + Translation + Scaling)

scale = 1.2  # Scale the image by 20%

M_similarity = cv2.getRotationMatrix2D(center, angle, scale)

similarity_transformed_image = cv2.warpAffine(image, M_similarity, (image.shape[1],
image.shape[0]))

display_image("Similarity Transformation (Rotation + Translation + Scaling)",
similarity_transformed_image)


# 4. Affine Transformation (Allows rotation, translation, and shearing)

# Get the image size (width and height)

rows, cols = image.shape[:2]

M_affine = np.float32([

    [1.1, 0.1, 30],  # Horizontal scaling and shearing

    [0.1, 1.1, 50]   # Vertical scaling and shearing

])

affine_transformed_image = cv2.warpAffine(image, M_affine, (cols, rows))

display_image("Affine Transformation", affine_transformed_image)
```