40923129L 湯可伊

Homework 5 (Due: 4/13)

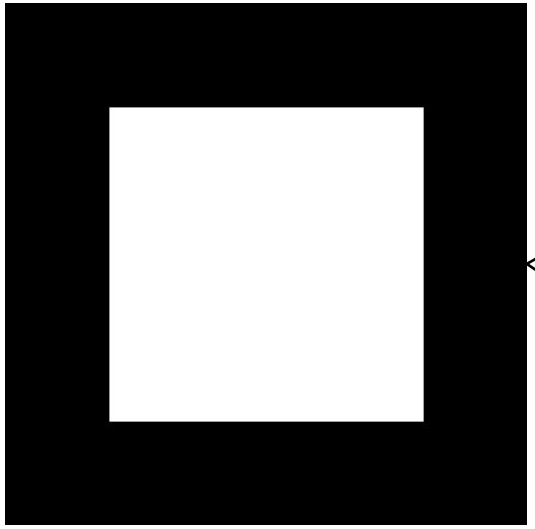
(1) Create an image consisting of a white square with a black background, e.g.,

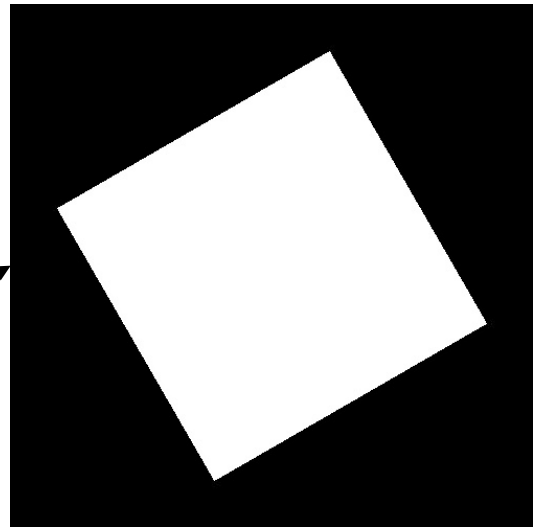
(2) Rotate the image by 30 degrees. Use (a) rotation with neighbor interpolation, and (b) rotation with bilinear interpolation.

(3) Compare the two results.

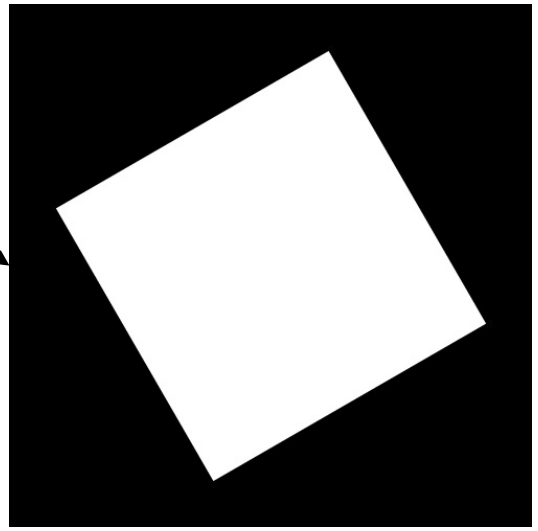
neighbor interpolation

input



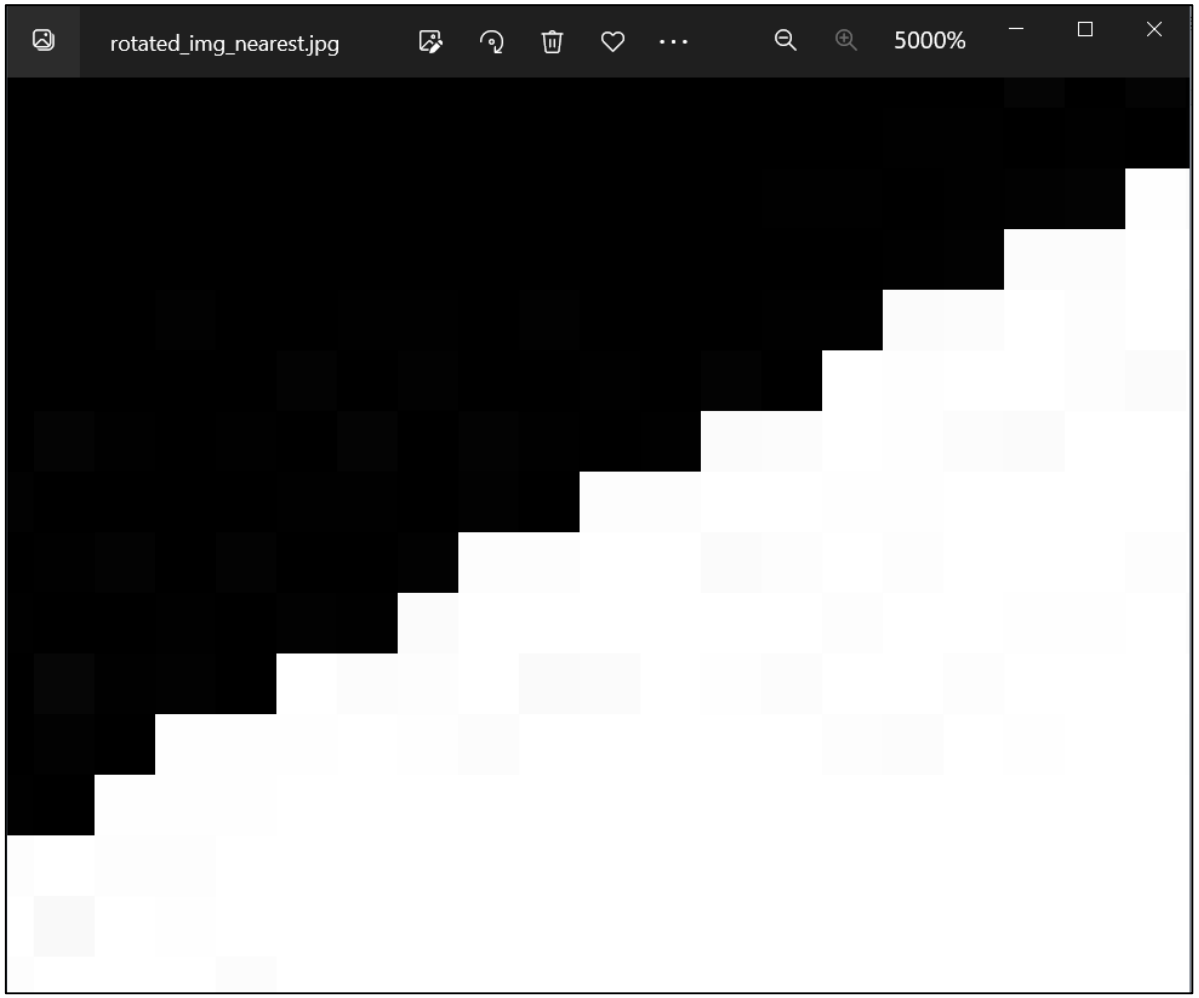


bilinear interpolation

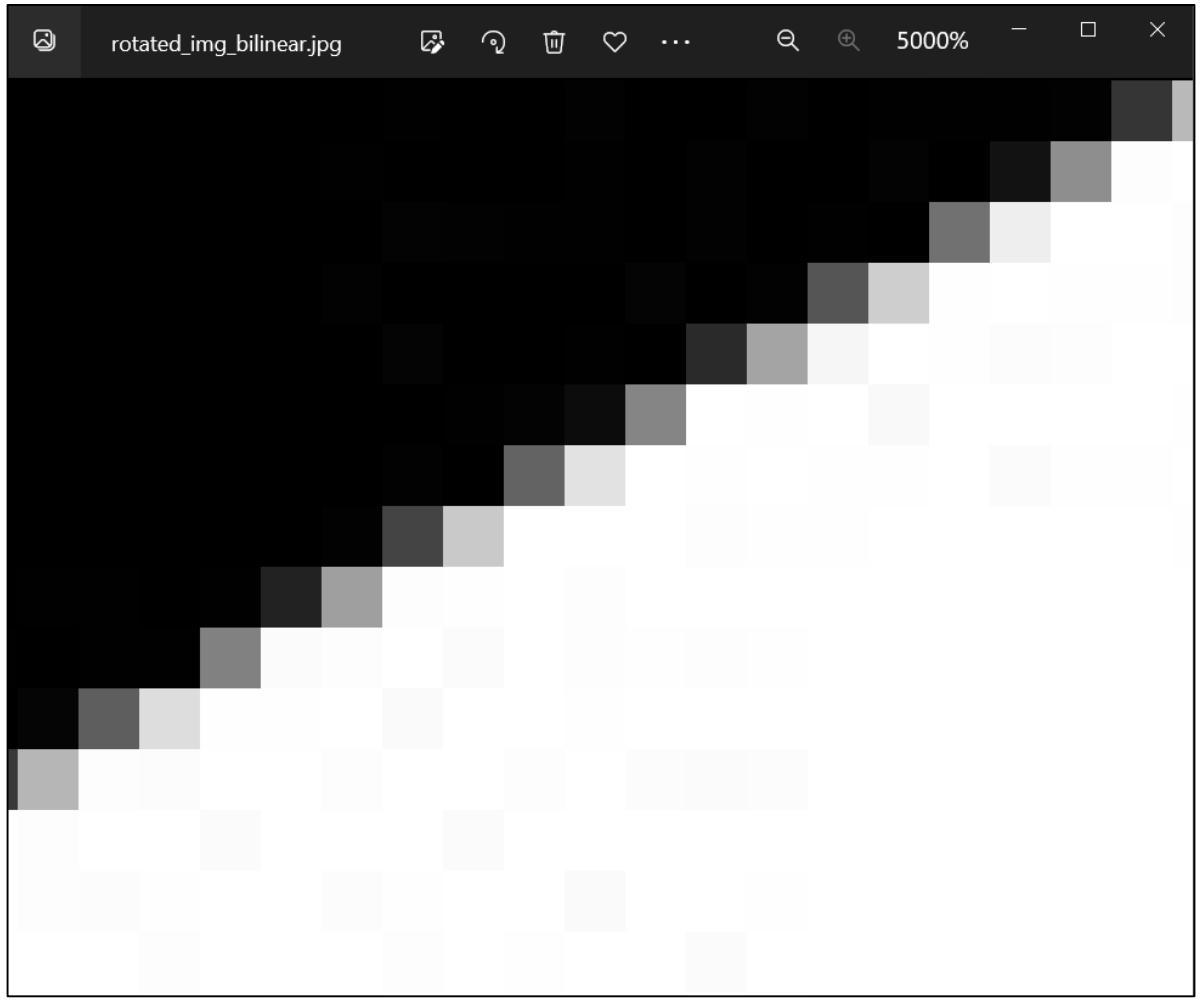


X5000%

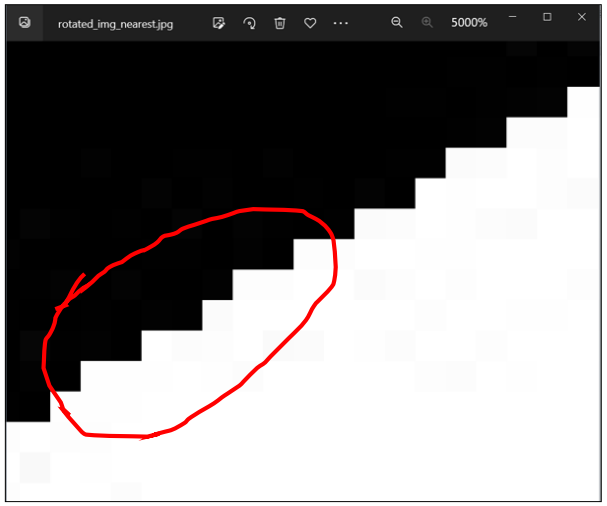
neighbor interpolation



bilinear interpolation



neighbor interpolation



bilinear interpolation

Neighbor interpolation works by assigning the value of the nearest pixel in the original image to the corresponding pixel in the rotated image. So in the neighbor interpolation the edge of the image only has two colors. Because of that, the edge of the neighbor interpolation image is more jagged than the edge of the bilinear interpolation image

Bilinear interpolation works by taking into account the values of the surrounding pixels to compute a weighted average that represents the value of the rotated pixel. So in the bilinear interpolation the edge of the image has not only black and white but also many different gray. Because of that, the edge of the bilinear interpolation image is smoother than the edge of the neighbor interpolation image.

Source code

```
# python Assignment_5.py

import cv2
import numpy as np
from PIL import Image,ImageDraw,ImageFont

# Define image size
size = (500, 500)
# Create a new black image
img = Image.new("RGB", size, "black")
# Define the coordinates for the white square
x0, y0 = 100, 100
x1, y1 = 400, 400
# Draw a white rectangle on the black image
draw = ImageDraw.Draw(img)
draw.rectangle([x0, y0, x1, y1], fill="white")
# Save the image as "white_square.jpg"
img.save("white_square.jpg")
```

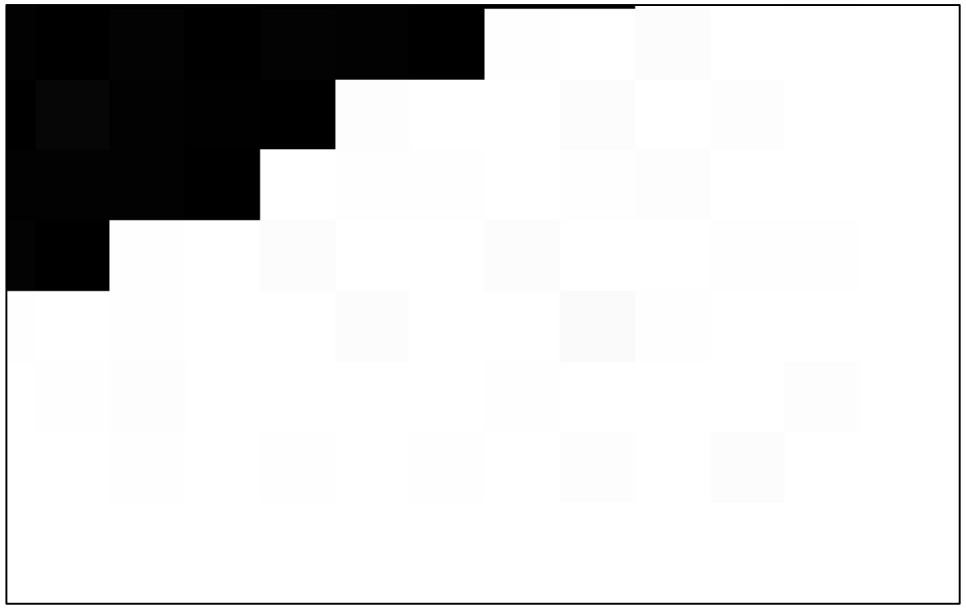
Source code

```
#-----(a) neighbor interpolation
img = cv2.imread('white_square.jpg')
# Determine the center of the image
(h, w) = img.shape[:2]
center = (w // 2, h // 2)
# Create rotation matrix for 30 degrees rotation
M = cv2.getRotationMatrix2D(center, 30, 1)
# Apply rotation to the image with neighbor interpolation
rotated_img_n = cv2.warpAffine(img, M, (w, h),
flags=cv2.INTER_NEAREST)
# Show image
cv2.imshow('rotated_img_nearest', rotated_img_n)
cv2.waitKey(0)
# Save the rotated image
cv2.imwrite('rotated_img_nearest.jpg', rotated_img_n)
```

```
#---------(b) bilinear interpolation
# Apply rotation to the image with bilinear interpolation
rotated_img_b = cv2.warpAffine(img, M, (w, h),
flags=cv2.INTER_LINEAR)
# Show image
cv2.imshow('rotated_img_bilinear', rotated_img_b)
cv2.waitKey(0)
# Save the rotated image
cv2.imwrite('rotated_img_bilinear.jpg', rotated_img_b)
```

Comments

I don’t know why there are some light gray squares in neighbor interpolation image. I think that neighbor interpolation get the value of a pixel from pixels near it, so there most be only black or white in the whole image. I guess that it is because we can not see every single pixel on screen. These gray squares are actually many black and white pixels.



neighbor interpolation image

original image