Activity 3: Image Manipulation

Contributors go to Keane Dalisay, Nel Alanan, and Prince Alexander Malatuba.

Cheers!

Handling single or multiple images

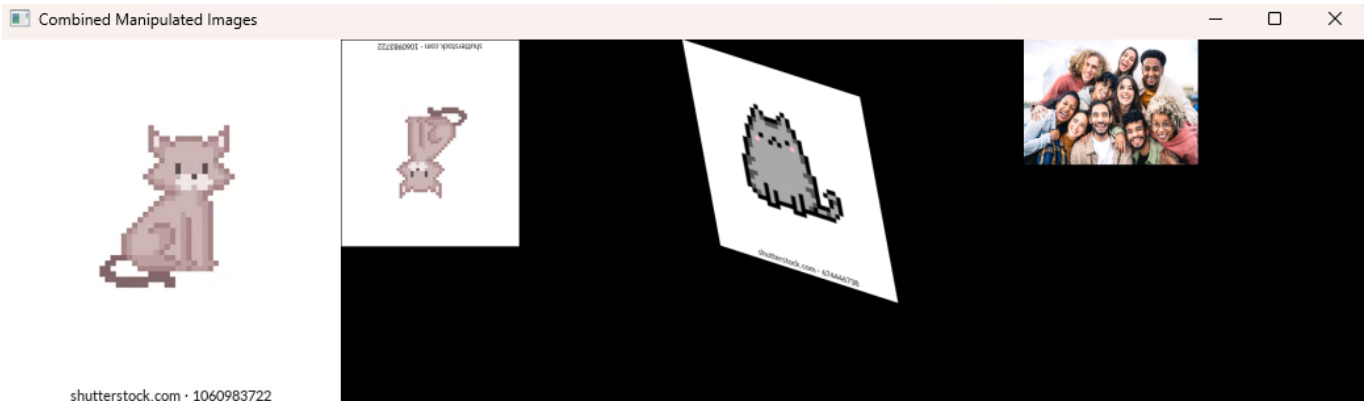
Functions were created for handling a number of images provided by the user.

```
def singleManip(self, img):
    manip_img = cv.imread(img.paths)
    running = True
    while running:
        img.displayImg('Original Image...', manip_img)
        manip_img = self.manipChoice(img, manip_img)
        img.displayImg('Manipulated Image...', manip_img)
        running = self.runAgain()
```

The original image before manipualtion.

```
def multiManip(self, imgs):
    fst_img = cv.imread(imgs.paths[0])
    running = True

    while running:
        fst_img_wdth = fst_img.shape[1]
        fst_img_hght = fst_img.shape[0]
        comb_img = fst_img
        i = 0
```

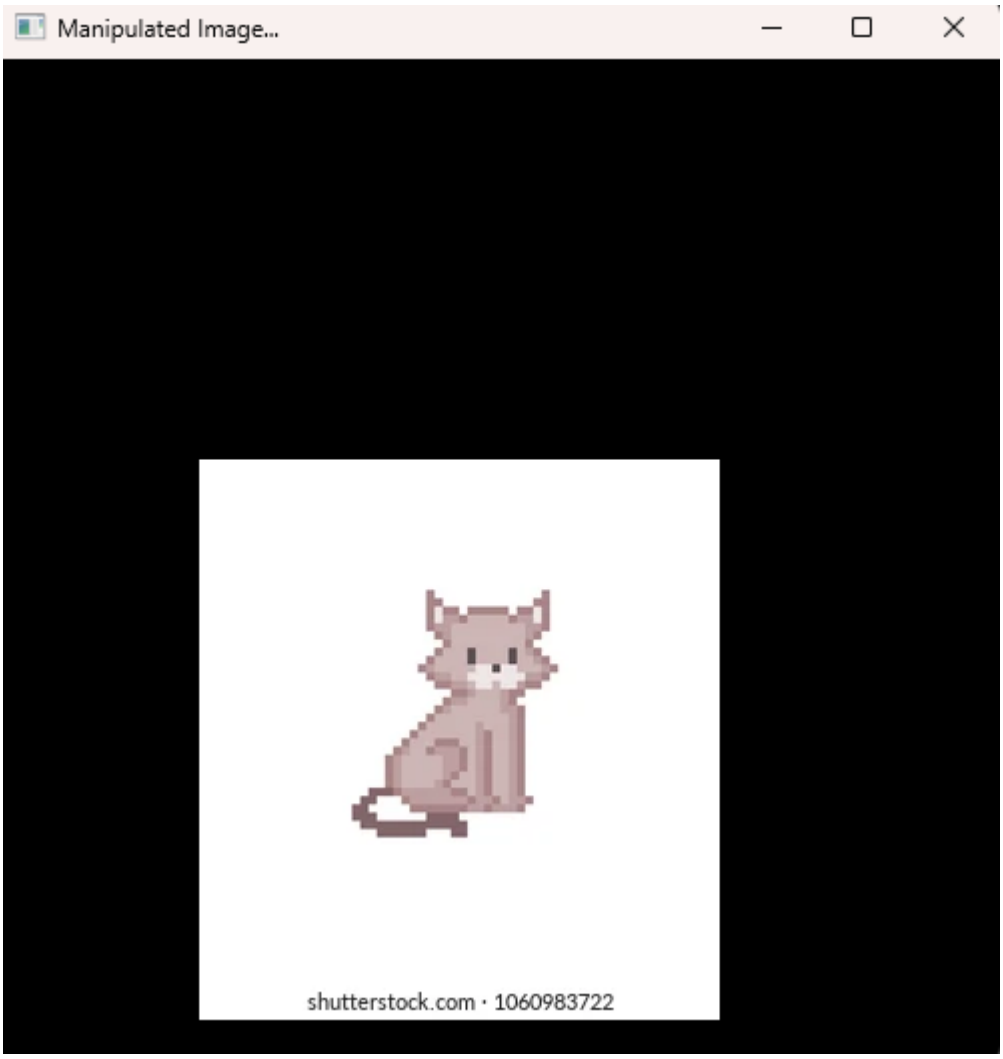
```
        while i < len(imgs.paths):
            trgt_img = cv.imread(imgs.paths[i])

            imgs.displayImg('Original Image...', trgt_img)
            manip_img = self.manipChoice(imgs, trgt_img)
            imgs.displayImg('Manipulated Image...', manip_img)

            resized_img = cv.resize(manip_img,
            (fst_img_wdth, fst_img_hght),
            interpolation = cv.INTER_AREA)

            comb_img = np.concatenate((comb_img, resized_img),axis=1)
            i += 1

        imgs.displayImg('Combined Manipulated Images', comb_img)
        running = self.runAgain()
```



Shows all images that have been manipulated.

Translating Images

```
def translate(img):
    wdth = img.shape[1]
    hght = img.shape[0]

    x = int(input('\nPixels to translate horizontally: '))
    y = int(input('\nPixels to translate vertically: '))

    m_translation = np.float32([
        [1, 0, x],
        [0, 1, y],
        [0, 0, 1]
    ])

    translated_img = cv.warpPerspective(img, m_translation, (mi(wdth * 2, 500),
min(hght * 2, 500)))
    return translated_img
```



A cat image translated 100 pixels to the left and 50 pixels downward.

Rotating Images

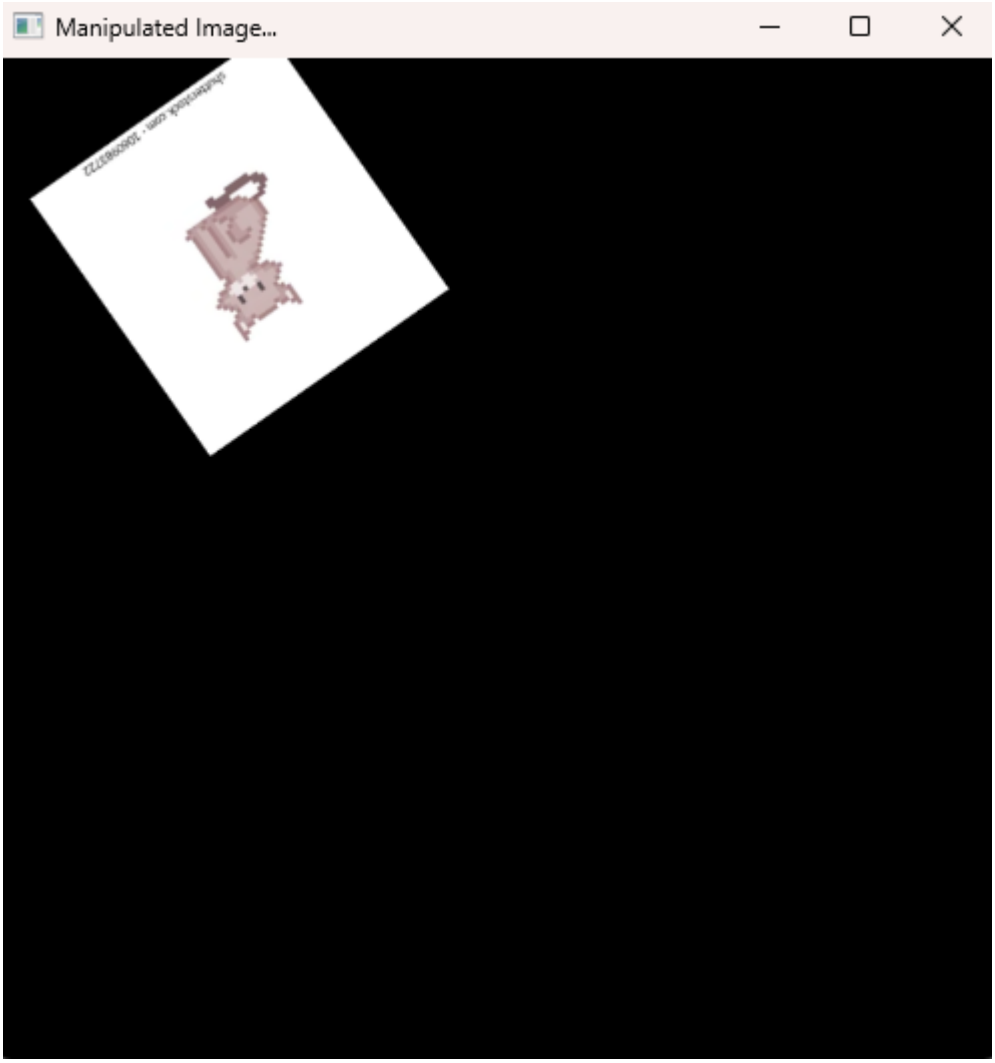
```
def rotate(img):
    angle = int(input('\nAngle to rotate image: '))

    wdth = img.shape[1]
    hght = img.shape[0]

    m_translation = cv.getRotationMatrix2D((wdth / 2, hght /2), angle, 1)
    rotated_img = cv.warpAffine(img, m_translation, (min(wdth *2, 500), min(hght *
2, 500)))

    return rotated_img
```

A cat image rotated 125 degrees counter-clockwise.

Scaling / Resizing Images

```
def scale(img):
    wdth = img.shape[1]
    hght = img.shape[0]

    print('\nScale percentage of width in decimals? ')
    scale_wdth = float((input(': ')))

    print('\nScale percentage of height in decimals? ')
    scale_hght = float((input(': ')))

    m_scaling = np.float32([
        [scale_wdth, 0, 0],
        [0 , scale_hght, 0],
        [0, 0, 1]
    ])

    scaled_img = cv.warpPerspective(img, m_scaling, (min(wdth *2, 500), min(hght *
2, 500)))
    return scaled_img
```

```
def resize(img):
    resize_wdth = int(input('\nPixels to resize image width: '))
    resize_hght = int(input('\nPixels to resize image height:'))

    resized_img = cv.resize(img, (resize_wdth,
resize_hght),interpolation=cv.INTER_AREA)
    return resized_img
```



A cat image scaled to 50% of its width and height.

Reflecting / Flipping Images

```
def reflection(self, img):
    wdth = img.shape[1]
    hght = img.shape[0]

    reflect_hoz = self.reflectChoice('\nFlip imagehorizontally? ')
    reflect_ver = self.reflectChoice('\nFlip image vertically?')

    m_reflection_ = np.float32([
```

```
        [reflect_hoz, 0, wdth],
        [0, reflect_ver, hght],
        [0, 0, 1]
    ])

    reflected_image = cv.warpPerspective(img, m_reflection_,(mi(wdth * 2, 500),
min(hght * 2, 500)))

    return reflected_image
```

A cat image reflected/flipped the opposite way horizontally and vertically.

Sheering / Skewing Images

```
def skew(img):
    wdth = img.shape[1]
    hght = img.shape[0]

    print('\nPercentage to skew horizontally in decimals? ')
    skew_hoz = float(input(': '))

    print('\nPercentage to skew vertically in decimals? ')
    skew_ver = float(input(': '))
```

```
    m_skewing = np.float32([
        [1, skew_hoz, 0],
        [skew_ver, 1, 0],
        [0, 0, 1]
    ])

    skewed_img = cv.warpPerspective(img, m_skewing, (min(wdth *2, 500), min(hght *
2, 500)))

    return skewed_img
```

A cat image skewed horizontally 100% to the right.