Computer Vision

Assignment 1

Pragya Jaiswal (MDS202129)

January 16, 2023

Problem 1:

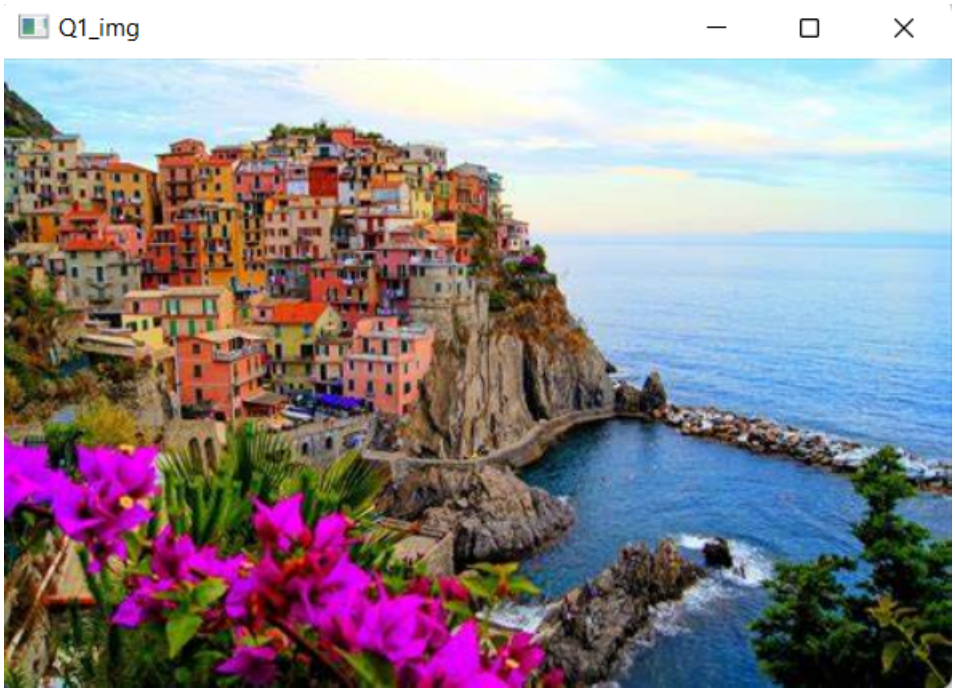
Part(a)

```
import cv2 as cv
import numpy as np

#Reading the image
file = 'CinqueTerre.jpg'
img = cv.imread(file)

#Finding the required measurements
width = img.shape[1]
height = img.shape[0]
channels = img.shape[2]

print("The size of the image is: height = ",height ,", width = ",width)
print("There are",channels,"channels in the image. ")
```

```
The size of the image is: height =  315 , width =  474
There are 3 channels in the image.
```

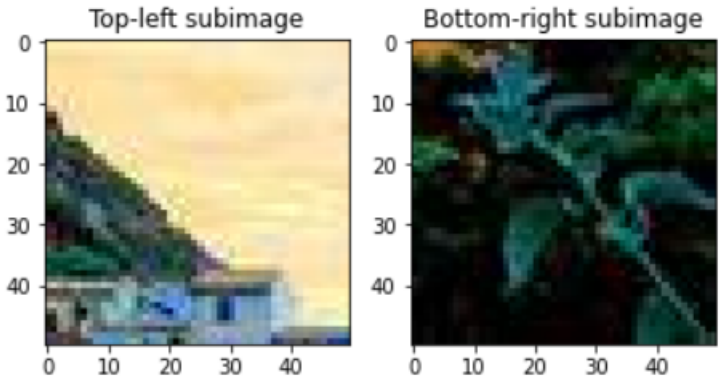
Displaying the image:

```
#Displaying the image
cv.imshow('Q1_img', img)
cv.waitKey(0)
cv.destroyAllWindows()

cv.imwrite("Q1_CinqueTerre.jpg",img)
```



Part(b)

```
#Creating the sub-images

subimg1 = img[0:50,0:50] #Top left subimage
subimg2 = img[265:315,424:474] #Bottom-right subimage

fig, ax = plt.subplots(1,2)
ax[0].imshow(subimg1)
ax[0].set_title('Top-left subimage')
ax[1].imshow(subimg2)
ax[1].set_title('Bottom-right subimage');
```



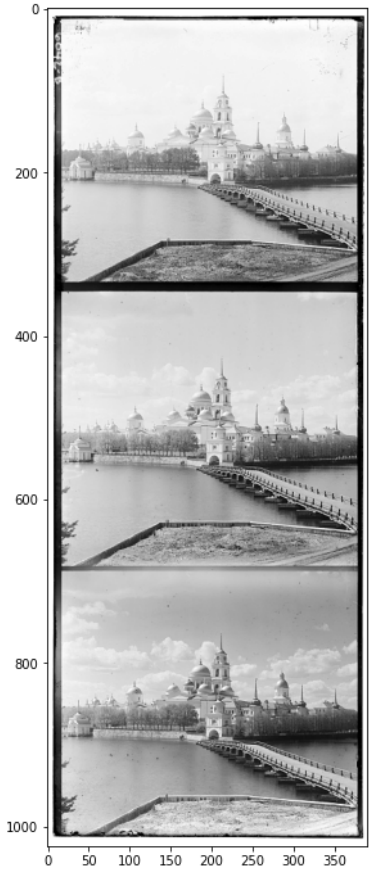

```
# Converting to grayscale for intensity
gray_1 = cv.cvtColor(subimg1, cv.COLOR_BGR2GRAY)
gray_2 = cv.cvtColor(subimg2, cv.COLOR_BGR2GRAY)
```

```
# Define function for ssd
calculate_ssd = lambda x,y: ((x.astype(int)-y.astype(int))**2).sum()

# Calculating SSD
SSD =  calculate_ssd(gray_1, gray_2)
print('SSD of the intensities = ', SSD)
```

```
SSD of the intensities =  62657575
```

Problem 2:

```
# Reading the image (Flag =0 for grayscale image)
img = cv.imread('monastery.jpg', 0)
plt.imshow(img, cmap='gray');
```




```
# Splitting the image vertically to obtain the three channels
b, g, r = np.array_split(img, indices_or_sections=3, axis=0)
print('Dimensions of the individual channels are:', (b.shape, g.shape, r.
```

```
Dimensions of the individual channels are:
((342, 391), (341, 391), (341, 391))
```

```
# Fixing the resolution of final image as that of the blue channel
resolution = b.shape

def padding(img, res):
  del_v = res[0] - img.shape[0]
  pad_t = del_v//2
  pad_b = del_v - pad_t
  del_h = res[1] - img.shape[1]
  pad_l = del_h//2
  pad_r = del_h - pad_l
  return cv.copyMakeBorder(img, pad_t, pad_b, pad_l, pad_r, cv.BORDER_CON


# Making all channels of the same resolution
g = padding(g, resolution)
r = padding(r, resolution)
```

Part(a)

```
# Function for sliding picture by (dx, dy)
def translate(img, dy, dx):
  # y-axis roll
  new_img = np.roll(img, dy, axis=0)
  if dy>0:
    new_img[0:dy,:]=0
  elif dy<0:
    new_img[dy:,:]=0

  # y-axis roll
  new_img = np.roll(new_img, dx, axis=1)
  if dx>0:
    new_img[:, 0:dx]=0
  elif dx<0:
    new_img[:, dx:]=0

  return new_img

# Slide picture and minimise SSD

def align_image(bottom, top):
  min_ssd, align, disp = np.inf, None, None
  for i in range(-10, 11):
```

```
    for j in range(-10, 11):
      new_top = translate(top, i, j)
      """
       Removing 10 pixels from each side of both images before calculatin
       SSD to remove noise at the borders
      """
      crop_top = new_top[10:new_top.shape[0]-10, 10:new_top.shape[1]-10]
      crop_bottom = bottom[10:bottom.shape[0]-10, 10:bottom.shape[1]-10]
      SSD = calculate_ssd(crop_bottom, crop_top)
      if SSD <= min_ssd:
        align = new_top
        min_ssd = SSD
        disp = [i, j]
  return align, disp, min_ssd
```

Part(b)

```
g_aligned, disp_g, ssd_g = align_image(b, g)
r_aligned, disp_r, ssd_r = align_image(b, r)

print(f'Green Channel:\ndisp= {disp_g}\t SSD= {ssd_g}')
print(f'Red Channel:\ndisp= {disp_r}\t SSD= {ssd_r}')
```

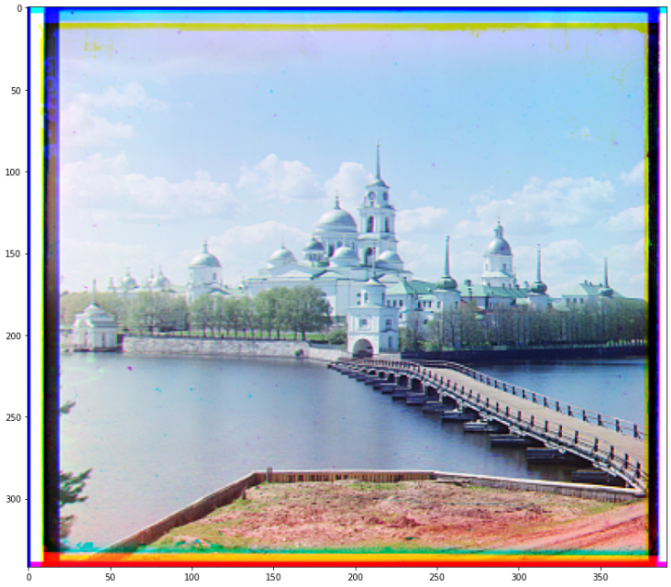
```
Green Channel:
Displacement= [-2, 2]    SSD= 169101693
Red Channel:
Displacement= [4, 2]     SSD= 392475460
```

Part(c)

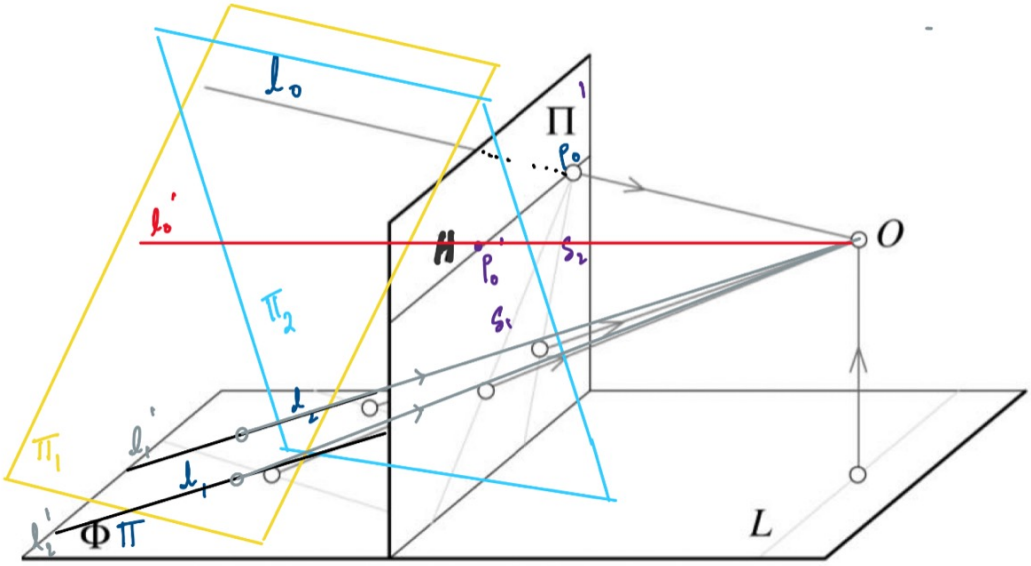
```
col_img = cv.merge([r_aligned, g_aligned, b])
```

Part(d)

```
plt.imshow(col_img);
```



Problem 3: Parallel lines appear to converge at horizon



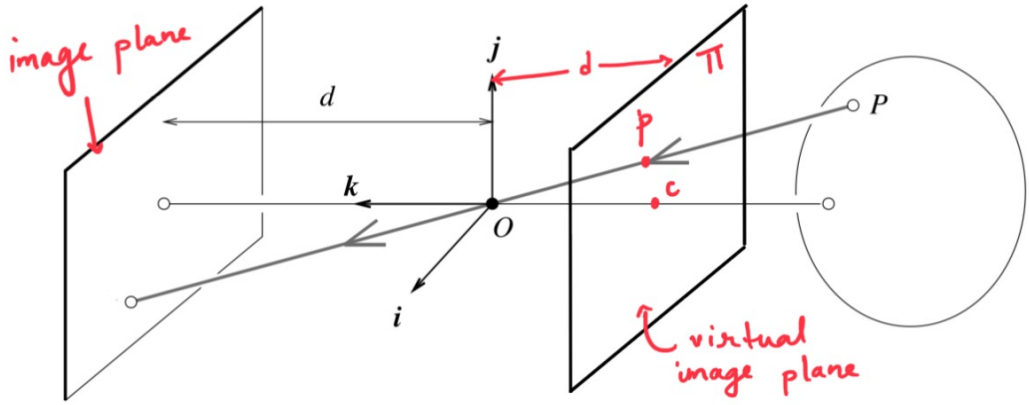
Let us consider two parallel lines l_1 and l_2 lying in the plane Π and define l_0 as the line passing through the pinhole that is parallel to l_1 and l_2.

The lines l_0 and l_1 define a plane Π_1, and the lines l_0 and l_2 define a second plane Π_2.

Clearly, $\Pi 1$ and $\Pi 2$ project onto the lines δ_1 and δ_2 where Π_1 and Π_2 intersect the image plane Π'.
These two lines intersect at the point p_0 where l_0 intersects Π'. This point is the vanishing point associated with the family of lines parallel to l_0, and the projection of any line in the family appears to converge on it. (This is true even for lines parallel to l_0 that do not lie in Π.)

Now let us consider two other parallel lines l_1' and l_2' in Π and define as before the corresponding line l_0' and vanishing point p_0'.
The lines l_0 and l_0' lie in a plane parallel to Π that intersects the image plane along a line H passing through p_0 and p_0'.
This is the horizon line, and any two parallel lines in Π appears to intersect on it. They appear to converge there since any image point above the horizon is associated with a ray issued from the pinhole and pointing away from Π.
Horizon points correspond to rays parallel to Π and points in that plane located at an infinite distance from the pinhole.

Problem 4: Derivation of Perspective Equations



Let us consider a coordinate system (O, i, j, k) attached to the pinhole camera whose origin O coincides with the pinhole.
Vectors i, j form the basis vectors of the plane parallel to the plane of the image.
Let the plane of virtual image formation be Π. It is located at a distance d in front of the pinhole along $-k$.
The line perpendicular to Π passes through O and c (image center) in Π. Let,
P - scene point with coordinates (X, Y, Z)
p - image of P with coordinates (x, y, z)
$z = -d$ (as p lies in the image plane in front of the pinhole (Origin))

As P, p, O are collinear, $\vec{Op} = \lambda \vec{OP}$
so,
$x = \lambda X$
$y = \lambda Y$
$z = \lambda Z$

$\implies \lambda = \frac{x}{X} = \frac{y}{Y} = \frac{-d}{Z}$

Hence,

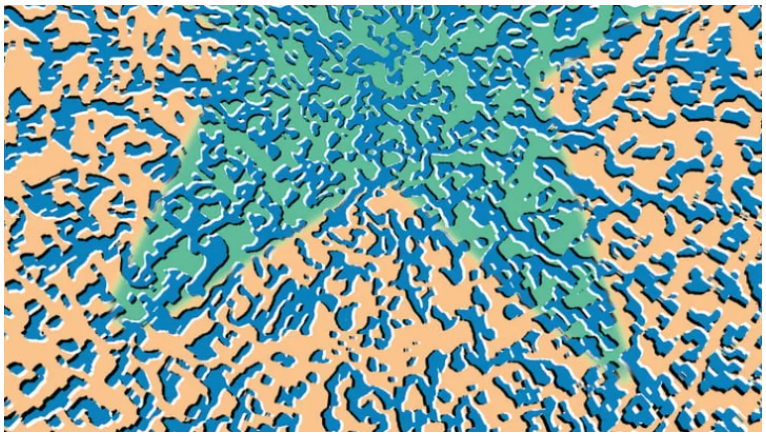
$x = \frac{-dX}{Z}$

$y = \frac{-dY}{Z}$

We have derived the perspective equations.

Problem 5: Illusions

1. **"Floating Star," Joseph Hautman/Kaia Nao, 2012 Finalist**



This five-pointed star is static, but many observers experience the powerful illusion that it is rotating clockwise. Created by the artist Joseph Hautman, who moonlights as a graphic designer under the pseudonym "Kaia Nao," it is a variation on Kitaoka's Rotating Snakes Illusion. Hautman determined that an irregular pattern, unlike the geometric one Kitaoka used, was particularly effective for achieving illusory motion.
Here the dark blue jigsaw pieces have white and black borders against a lightly colored background. As you look around the image, your eye movements stimulate motion-sensitive neurons. These neurons signal motion by virtue of the shifting lightness and darkness boundaries that indicate an object's contour as it moves through space. Carefully arranged transitions between white, light-colored, black, and dark-colored regions fool the neurons into responding as if they were seeing continual motion in the same direction, rather than stationary edges.

2. **"Ghostly Gaze," Rob Jenkins // University of Glasgow, UK, 2008 Second Prize**

Not knowing where a person is looking makes us uneasy. That's why speaking with somebody who is wearing dark sunglasses can be awkward. And it is why someone might wear dark sunglasses to look "mysterious." The Ghostly Gaze Illusion, created by Rob Jenkins, takes advantage of this unsettling effect. In this illusion, twin sisters appear to look at each other when seen from afar. But as you approach them, you realize that the sisters are looking directly at you!
The illusion is a hybrid image that combines two pictures of the same woman. The overlapping photos differ in two important ways: their spatial detail (fine or coarse) and

the direction of their gaze (sideways or straight ahead). The images that look toward each other contain only coarse features, whereas the ones that look straight ahead are made up of sharp details. When you approach the pictures, you are able to see all the fine detail, and so the sisters seem to look straight ahead. But when you move away, the gross detail dominates, and the sisters appear to look into each other's eyes.