Solution 1

In [1]:
```python
#importing the libraries
import cv2 as cv
import numpy as np
import matplotlib.pyplot as plt
```

In [24]:
```python
img = cv.imread('CinqueTerre.jpg')
plt.imshow(cv.cvtColor(img, cv.COLOR_BGR2RGB))
```

Out[24]:
```
<matplotlib.image.AxesImage at 0x266402bd490>
```

a)

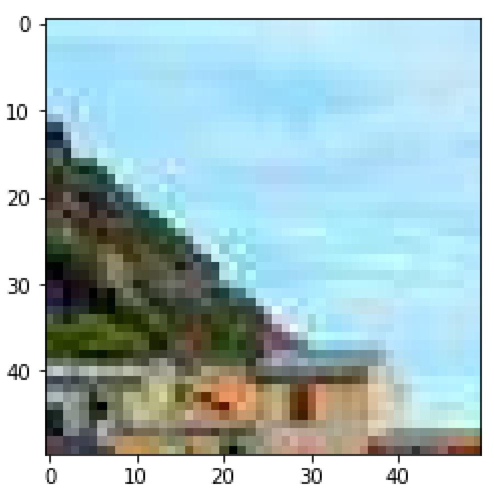
In [3]:
```python
#Size and Number of channels of the image
print("The size of the image is:", img.shape)
print("Number of channels in the image is: ", img.shape[-1])
```

```
The size of the image is: (315, 474, 3)
Number of channels in the image is:  3
```

b)

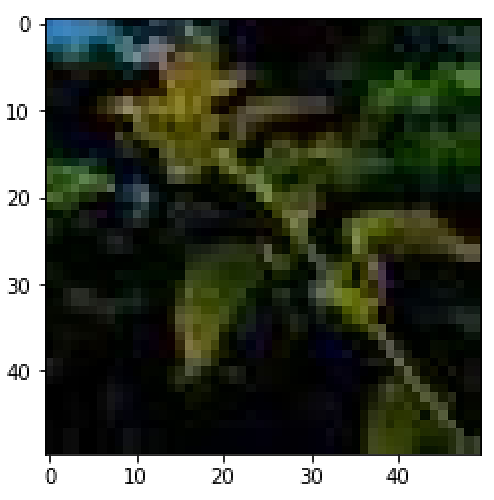
In [21]:
```python
#getting the top left part of the image
subimg1 = img[:50,:50,:]
print(subimg1.shape)
plt.imshow(cv.cvtColor(subimg1, cv.COLOR_BGR2RGB))
plt.show()
```

```
(50, 50, 3)
```



In [22]:
```python
#getting the bottom right part of the image
subimg2 = img[265:,424:,:]
print(subimg2.shape)
plt.imshow(cv.cvtColor(subimg2, cv.COLOR_BGR2RGB))
plt.show()
```

```
(50, 50, 3)
```



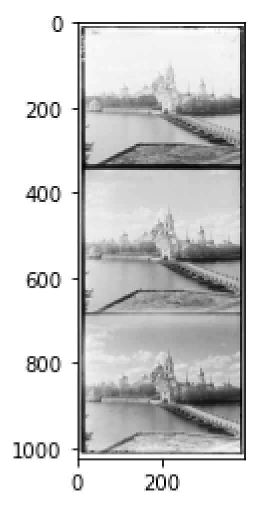
In [25]:
```python
#Funtion to calculate SSD
def ssd(img1, img2):
    diff = img1 - img2
    ssd = np.sum(diff**2, axis=(0,1)).sum()
    return ssd
```

In [27]:
```python
#Calculating the ssd of the bottom right and top left part
SSD = ssd(subimg1,subimg2)
print("The SSD of the two images is : ", SSD)
```

```
The SSD of the two images is :  781484
```

Solution 2

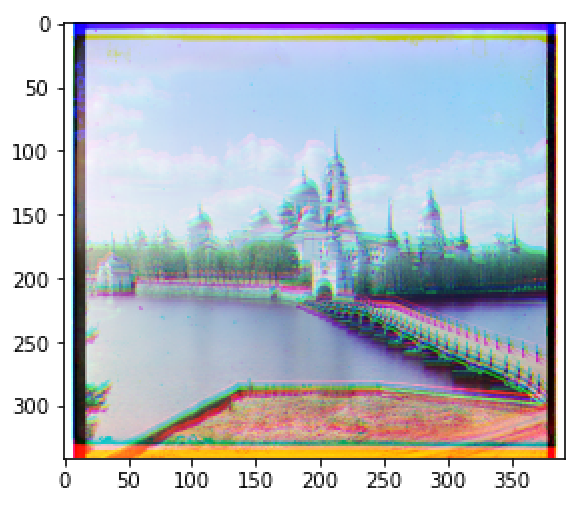
In [30]:
```python
#reading the image as grayscale
img2 = cv.imread('monastery.jpg', cv.IMREAD_GRAYSCALE)
plt.imshow(cv.cvtColor(img2, cv.COLOR_BGR2RGB))
plt.show()
```



In [31]:
```python
#splitting the image into blue, green and red channels
w,h=img2.shape
height=int(w/3)
blue=img2[0:height]
green=img2[height:2*height]
red=img2[2*height:3*height]
size = img2.shape
print("The size of the image is :", size)
```

```
The size of the image is : (1024, 391)
```

In [32]:
```python
#image without alignment
merged = cv.merge([blue,green,red])
plt.imshow(cv.cvtColor(merged, cv.COLOR_BGR2RGB))
plt.show()
```



a)

In [87]:
```python
#function to crop the image
def crop(img):
    height, width = img.shape
    new_height, new_width = int(height/1.11), int(width/1.11)
    return img[:new_height, :new_width]
```

In [88]:
```python
#function to slide the window over the base image
def slide1(img1,img2):
    img1_sub = crop(img1)
    h,w = img1_sub.shape
    min_ssd = ssd(img1,img2)
    for i in range(10):
        for j in range(10):
            img2_sub = img2[i:i+h,j:j+w]
            ssd1 = ssd(img1_sub,img2_sub)
            if ssd1<min_ssd:
                min_ssd = ssd1
                best_img = [i,j]
    return(min_ssd,best_img)
```

a)

In [89]:
```python
#we shall fix the base image as green and slide red and blue over it
b_on_g = slide1(blue,green)
r_on_g = slide1(red,green)
g_fin = green
b_fin = np.roll(blue,(b_on_g[1][0],b_on_g[1][1]), axis =(0,1))
r_fin = np.roll(red,(r_on_g[1][0],r_on_g[1][1]), axis =(0,1))
```

b)

In [83]:
```python
print("Best SSD value and displacement vector for blue on green are",b_on_g)
print("Best SSD value and displacement vector for red on green are",r_on_g)
```

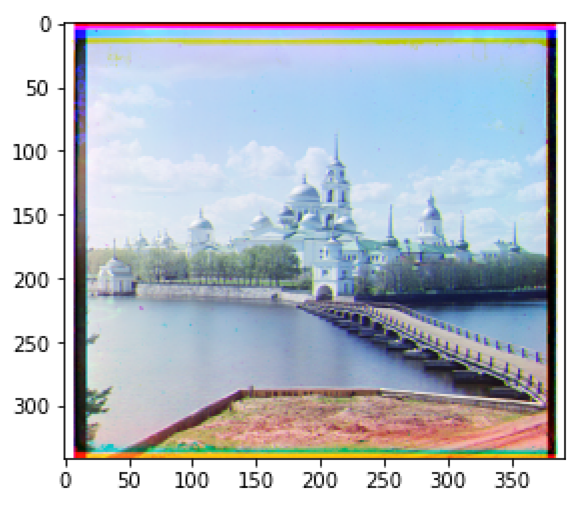
```
Best SSD value and displacement vector for blue on green are (11115572, [3, 0])
Best SSD value and displacement vector for red on green are (10611427, [6, 1])
```

c)

In [84]:
```python
#merging the best aligned channels
merged = cv.merge([b_fin,g_fin,r_fin])
```

d)

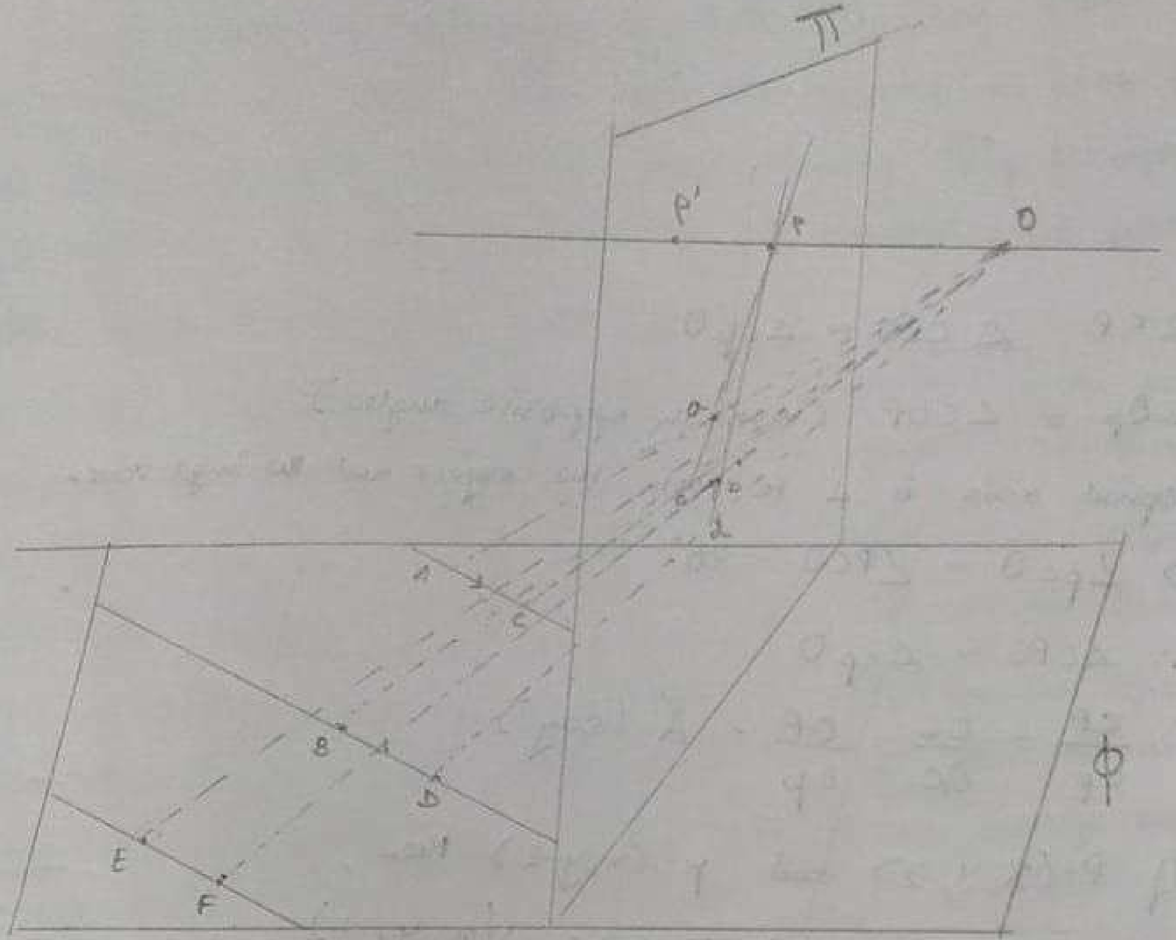
In [85]:
```python
#displaying the merged image
plt.imshow(cv.cvtColor(merged, cv.COLOR_BGR2RGB))
plt.show()
```



In [ ]:

CV Assignment 1

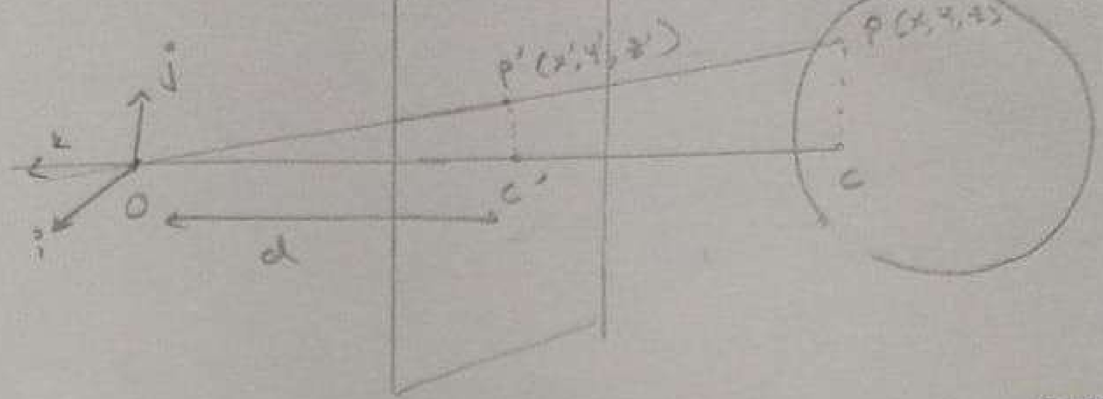
3)



Given a pair of parallel lines if we join them to the pinhole we get the dashed straight lines.
Now, if we place a virtual screen between the ~~lines~~ lines and the pinhole we can get the corresponding image of the parallel lines.
So A will go to a & C will go to c, we get a corresponding virtual image of AC. Similarly for BC we get DC. On extending these lines on the screen, they intersect at P, the vanishing point.

If we take any other pair ~~the~~ corresponding pair intersects at say P' on the virtual screen. Then we can see, P, P' & O lie on a straight line. This line is horizon line h and it is the intersection of a plane ∥ to ϕ and π & passes through the pinhole.

4)



Let OC be the optical axis & P be the object. When we look at the virtual image of P we have P'.

$OC' = d$.

R.T.P: $\Delta OP'C' \sim \Delta OPC$

$\angle P'OC' = \angle POC$ (common)

$\angle P'C'O = \angle PCO = 90^\circ$ (the virtual image screen and the object screen are perpendicular to the optical axis)

$\therefore \Delta OP'C' \sim \Delta OPC$ (AA axiom)

$\Rightarrow \frac{OP'}{OP} = \frac{OC'}{OC} = \frac{P'C'}{PC} = \lambda$ (say)

$\therefore OP' = \lambda OP \Rightarrow (x', y', z') = \lambda(x, y, z)$

$\Rightarrow x' = \lambda x,\ y' = \lambda y,\ z' = \lambda z$

Note that, $\hat{k}$ points towards the left, hence the right side has negative values. $\Rightarrow z' = -d$.

$\therefore \frac{x'}{x} = \frac{y'}{y} = \frac{-d}{z} \Rightarrow x' = -\frac{d}{z}x,\ y' = -\frac{d}{z}y,\ z' = -d$.

The perspective equation projection for a virtual image is,

$x' = -\frac{d}{z}x,\ y' = -\frac{d}{z}y,\ z' = -d$.

5) Two of the illusions are :

(a) Ghostly Gaze

This image has the photograph of ~~two~~ twin sisters. When seen initially it looks like they are looking at each other but when carefully observed, they actually look straightwards. This difference is seen because of two reasons, the spatial detail of the image and their gaze.
The image was created by combining two pictures of the same women, one which was coarse had the women looking towards each other. The other one which was fine had them looking straight. So when we see the image from a distance we can only see the rough i.e. coarse details hence sideways. Whereas, if we go close to the image we can see the fine details that is the women looking straightward.

(b) The Rotating Snakes Illusion

As we move ~~the~~ eyes over the image, it appears as if the spirals are moving. But when we fix our gaze, this movement slows down or even stops. This happens because of our jerky eye motions - such as rapid eye movements like microsaccades, larger saccades and blinks.
This image accelerates the motion-sensitive neurons in the visual cortex due to which our visual neurons make us perceive that the snakes are rotating.