AI6121- Computer Vision

Assignment 2

Brandon Chua Shaojie

G1903442H

Task 1: Describe the procedure of disparity computing given a pair of rectified images of the same scene captured from two different viewpoints.

Disparity is the difference in image location of the same 3D point under the perspective of two different cameras.

For each epipolar line,

1. To find the disparity, we match a patch of pixels in the left image with a patch of pixels in the right image along the horizontal epipolar line. This is assuming that the optical axes of the two cameras are parallel.

2. Pick the patch of pixels that has minimum cost using sum of squared differences:
$$\sum\sum\left(L(r,c) - R(r,c-d)\right)^2$$
$$\text{where } L \text{ and } R \text{ is the left and right image columns,}$$
$$r \text{ and } c \text{ is the current row and column,}$$
$$\text{and } d \text{ is the disparity of the right image}$$

3. Compute disparity, where disparity = (x coordinate of left pixel) - (x coordinate of right pixel)

Task 2: Write a computer algorithm that computes the disparity of two images. The programming language can be Matlab, python, or other languages.

<u>For corridor image:</u>

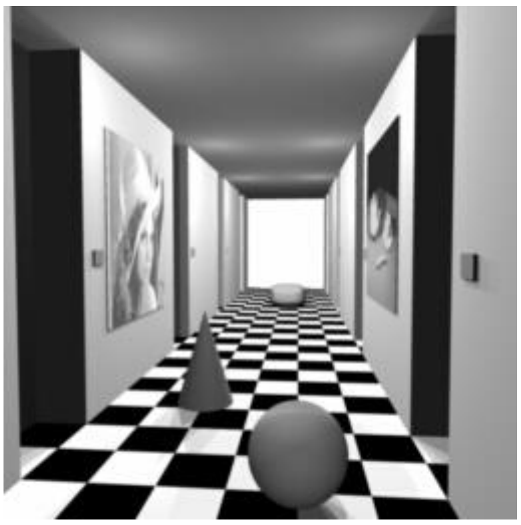
```
import numpy as np

import cv2

from matplotlib import pyplot as plt


imgL = cv2.imread('corridorl.jpg',0)

imgR = cv2.imread('corridorr.jpg',0)


stereo = cv2.StereoBM_create(numDisparities=16, blockSize=15)

disparity = stereo.compute(imgL,imgR)

plt.imshow(disparity,'gray')

plt.savefig('corridorMAP.jpg')

plt.show()
```

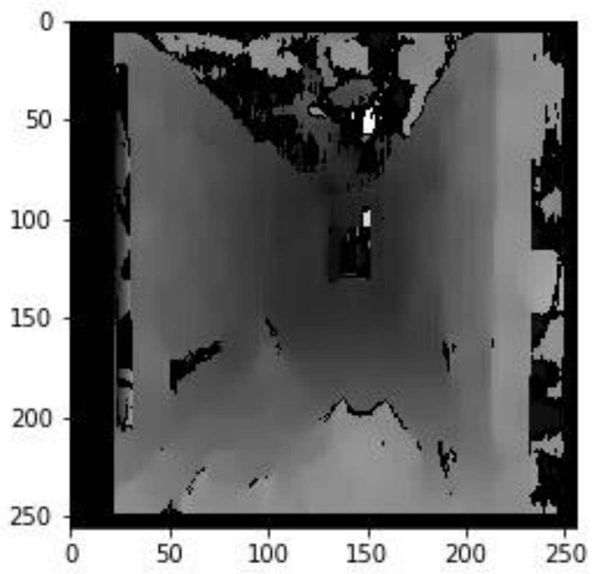
<u>For triclopsi21 image:</u>

```
imgL = cv2.imread('triclopsi2l.jpg',0)

imgR = cv2.imread('triclopsi2r.jpg',0)


stereo = cv2.StereoBM_create(numDisparities=16, blockSize=15)

disparity = stereo.compute(imgL,imgR)

plt.imshow(disparity,'gray')

plt.savefig('triclopsi2MAP.jpg')

plt.show()
```

Task 3: Apply your developed algorithm to the two provided image pairs and derive the corresponding disparity maps. Discuss your observation of the obtained disparity maps.

Corridor Image:

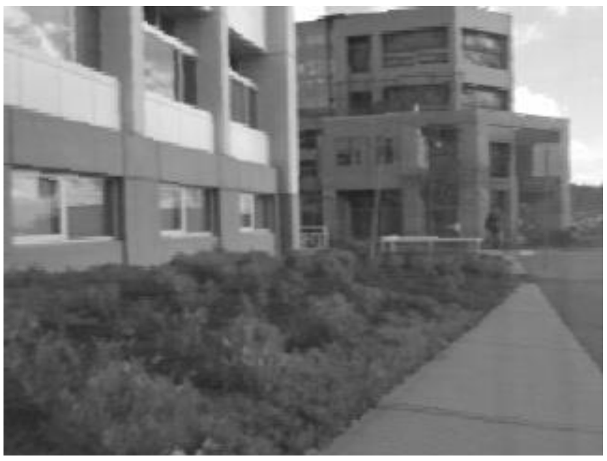


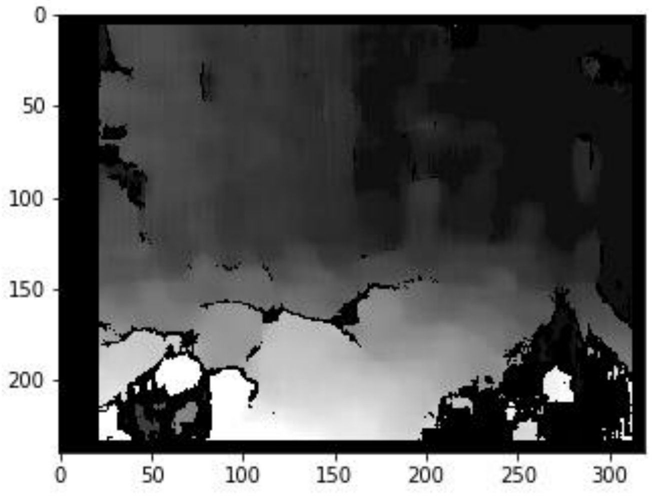


Brighter areas of the disparity map mean larger disparity and darker areas means smaller disparity. Since disparity is inversely proportionate to depth, brighter areas mean smaller depth and darker areas means larger depth. From the disparity map, the farther end of the hallway with small disparity is darker. The floor and walls of the corridor that are nearest to the viewer with large disparity is brightest.

Noise is observed on the sides and the ceilings. This is due to the homogenous surface.

Triclopsi2 Image:





From the disparity map, the farther building with small disparity is darker. The building that is nearer to the viewer with large disparity is brighter. The bushes nearest to the viewer is the brightest.

Noise is observed on a small part of the bushes. Almost the entire walkway is noisy. This is due to the homogenous surface.

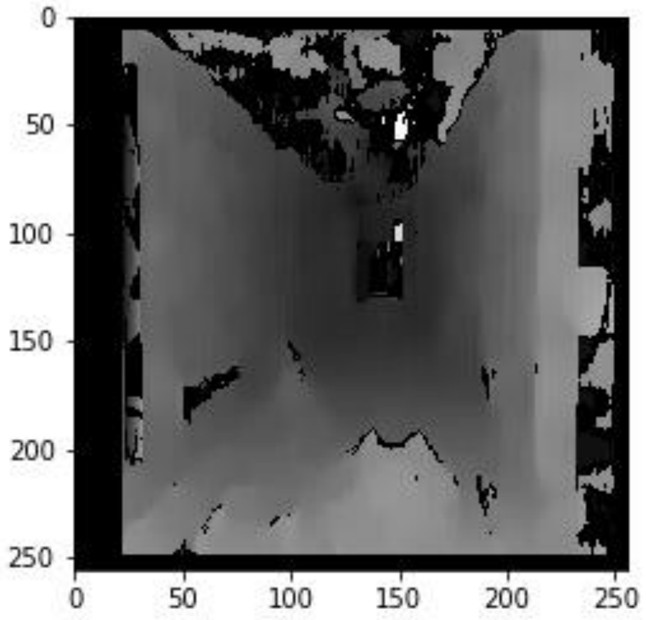
Task 4: Discuss what affects the disparity map computation, and any possible improvements of your developed algorithm. Implement and verify your ideas over the provided test images. This subtask is optional, and there will be credit marks for nice addressing of this subtask.

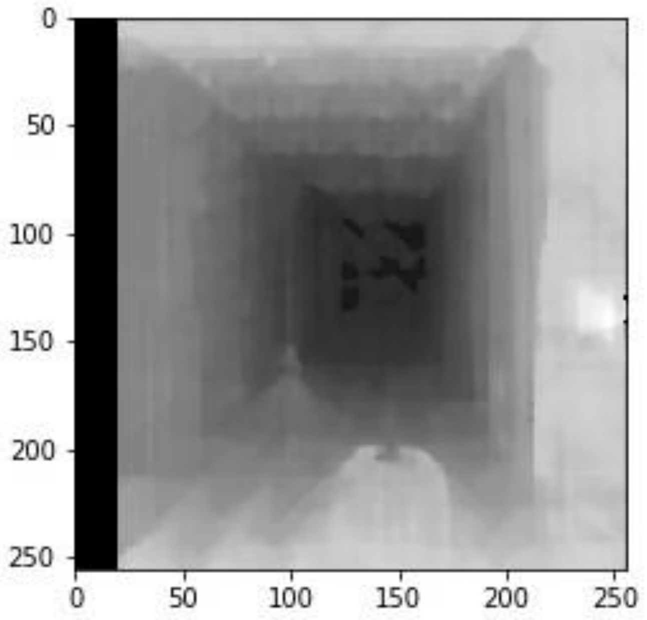
For computation, we can set the minimum and maximum disparities by adjusting the range along the epipolar line to find the matching patch of pixels. By restricting the search to a certain range, any matches outside of it will not be considered. This can reduce the chances of false matches.

Snippet of code (The entire code will be included in the appendix.):

```
min_disp = 1              <-----------------------------------------(Set minimum range)
num_disp = 20-min_disp    <----------------------------------------- (Set maximum range)
win_size = 5

stereo = cv2.StereoSGBM_create(minDisparity= min_disp,
 numDisparities = num_disp,
 blockSize = 5,
 uniquenessRatio = 5,
 speckleWindowSize = 5,
 speckleRange = 5,
 disp12MaxDiff = 1,
 P1 = 8*3*win_size**2,#8*3*win_size**2,
 P2 =32*3*win_size**2) #32*3*win_size**2)
```

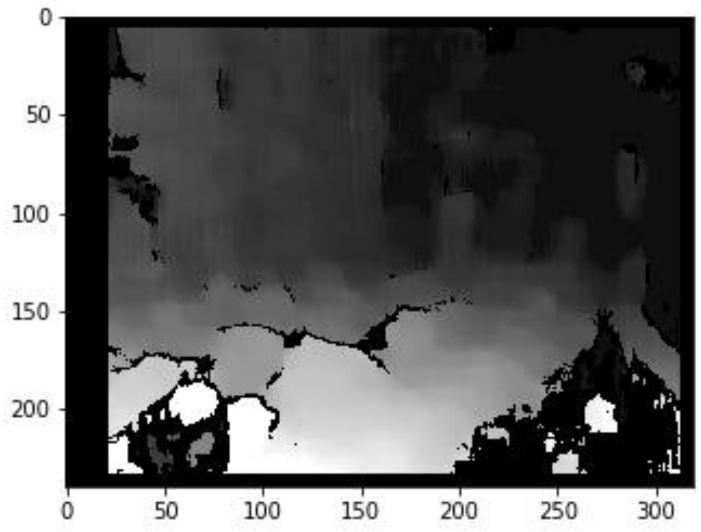
Corridor Image:

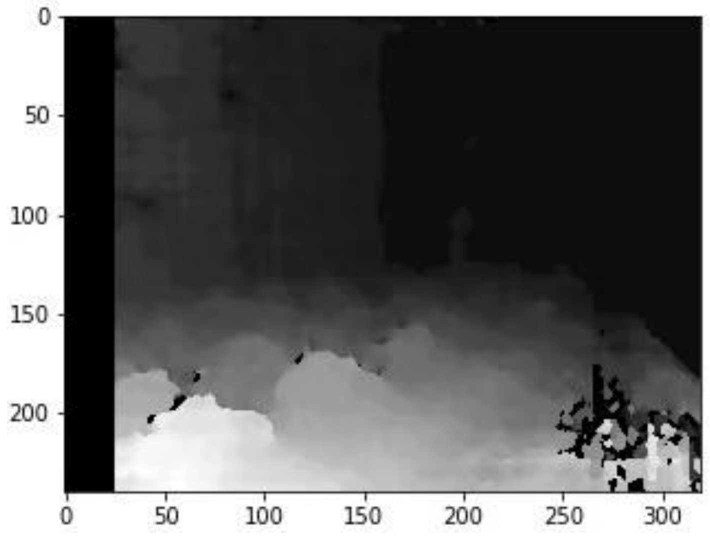




As seen, the disparity map on the right is smoother than the one on the left. This is especially so for the ceiling. With the new implemented algorithm of limiting the search range, the homogenous ceiling can be better handled, thus resulting in less noise.

Triclopsi2 Image:





Same with the case of the triclopsi image, the homogenous walkway can be better handled, thus resulting in much less noise. One more thing to note is that the noise of the nearest bush is also gone in the new disparity map. Overall, by limiting the search range, both images have resulted in better disparity maps that has less noise and handles homogenous surfaces better.

References

1. opencv/samples/python/stereo_match.py, Retrieved from https://github.com/opencv/opencv/blob/master/samples/python/stereo_match.py

2. Padierna O. (January 3 2019) Tutorial: Stereo 3D reconstruction with OpenCV using an iPhone camera. Part III. Retrieved from https://medium.com/@omar.ps16/stereo-3d-reconstruction-with-opencv-using-an-iphone-camera-part-iii-95460d3eddf0

3. Depth Map from Stereo Images, Retrieved from https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_calib3d/py_depthmap/py_depthmap.html

4. Corke P. (May 15, 2017) Computing Disparity, Retrieved from https://robotacademy.net.au/lesson/computing-disparity/

Appendix

```
import numpy as np

import cv2

from matplotlib import pyplot as plt


imgL = cv2.imread('corridorl.jpg',0)

imgR = cv2.imread('corridorr.jpg',0)


stereo = cv2.StereoBM_create(numDisparities=16, blockSize=15)

disparity = stereo.compute(imgL,imgR)

plt.imshow(disparity,'gray')

plt.savefig('corridorMAP.jpg')

plt.show()


imgL = cv2.imread('triclopsi2l.jpg',0)

imgR = cv2.imread('triclopsi2r.jpg',0)
```

```
stereo = cv2.StereoBM_create(numDisparities=16, blockSize=15)
disparity = stereo.compute(imgL,imgR)
plt.imshow(disparity,'gray')
plt.savefig('triclopsi2MAP.jpg')
plt.show()

imgL = cv2.imread('corridorl.jpg',0)
imgR = cv2.imread('corridorr.jpg',0)

min_disp = 1
num_disp = 20-min_disp
win_size = 5
stereo = cv2.StereoSGBM_create(minDisparity= min_disp,
 numDisparities = num_disp,
 blockSize = 5,
 uniquenessRatio = 5,
 speckleWindowSize = 5,
 speckleRange = 5,
 disp12MaxDiff = 1,
 P1 = 8*3*win_size**2,#8*3*win_size**2,
 P2 =32*3*win_size**2) #32*3*win_size**2)
disparity = stereo.compute(imgL,imgR)
plt.imshow(disparity,'gray')
plt.savefig('corridorMAPTuned.jpg')
plt.show()

imgL = cv2.imread('triclopsi2l.jpg',0)
imgR = cv2.imread('triclopsi2r.jpg',0)

min_disp = 1
num_disp = 25-min_disp
win_size = 5
```

```
stereo = cv2.StereoSGBM_create(minDisparity= min_disp,
 numDisparities = num_disp,
 blockSize = 5,
 uniquenessRatio = 5,
 speckleWindowSize = 5,
 speckleRange = 5,
 disp12MaxDiff = 1,
 P1 = 8*3*win_size**2,#8*3*win_size**2,
 P2 =32*3*win_size**2) #32*3*win_size**2)
disparity = stereo.compute(imgL,imgR)
plt.imshow(disparity,'gray'
plt.savefig('triclopsi2MAPTuned.jpg')
plt.show()
```

In [25]:

```python
import numpy as np
import cv2
from matplotlib import pyplot as plt

imgL = cv2.imread('corridorl.jpg',0)
imgR = cv2.imread('corridorr.jpg',0)

stereo = cv2.StereoBM_create(numDisparities=16, blockSize=15)
disparity = stereo.compute(imgL,imgR)
plt.imshow(disparity,'gray')
plt.savefig('corridorMAP.jpg')
plt.show()
```

...

In [6]:

```python
imgL = cv2.imread('triclopsi2l.jpg',0)
imgR = cv2.imread('triclopsi2r.jpg',0)

stereo = cv2.StereoBM_create(numDisparities=16, blockSize=15)
disparity = stereo.compute(imgL,imgR)
plt.imshow(disparity,'gray')
plt.savefig('triclopsi2MAP.jpg')
plt.show()
```

...

In [65]:

```python
imgL = cv2.imread('corridorl.jpg',0)
imgR = cv2.imread('corridorr.jpg',0)

min_disp = 1
num_disp = 20-min_disp
win_size = 5
stereo = cv2.StereoSGBM_create(minDisparity= min_disp,
 numDisparities = num_disp,
 blockSize = 5,
 uniquenessRatio = 5,
 speckleWindowSize = 5,
 speckleRange = 5,
 disp12MaxDiff = 1,
 P1 = 8*3*win_size**2,
 P2 =32*3*win_size**2)
disparity = stereo.compute(imgL,imgR)
plt.imshow(disparity,'gray')
plt.savefig('corridorMAPTuned.jpg')
plt.show()
```

...

In [68]:

```python
imgL = cv2.imread('triclopsi2l.jpg',0)
imgR = cv2.imread('triclopsi2r.jpg',0)

min_disp = 1
num_disp = 25-min_disp
win_size = 5
stereo = cv2.StereoSGBM_create(minDisparity= min_disp,
 numDisparities = num_disp,
 blockSize = 5,
 uniquenessRatio = 5,
 speckleWindowSize = 5,
 speckleRange = 5,
 disp12MaxDiff = 1,
 P1 = 8*3*win_size**2,
 P2 =32*3*win_size**2)
disparity = stereo.compute(imgL,imgR)
plt.imshow(disparity,'gray')
plt.savefig('triclopsi2MAPTuned.jpg')
plt.show()
```