CVE Assignment 7 Report

Question 1 (PS7)

Algorithm and Observations

The objective of this problem was to use the concept of stereo vision to calculate the depth information about two stereo images. As the camera matrix was not provided, the depth was modeled as a function of the disparity between the two images.

In block matching or cv2.StereoBM_create() the disparity is computed by comparing the sum of absolute differences (SAD) of each 'block' of pixels. In semi-global block matching or cv2.StereoSGBM_create() forces similar disparity on neighbouring blocks. This creates a more complete disparity map but is more computationally expensive.

Through trial and error with the ball set of images, a coefficient of 0.65 was found to work well in transforming the disparity to the depth. For calculating the disparity map, the values were chosen as follows:

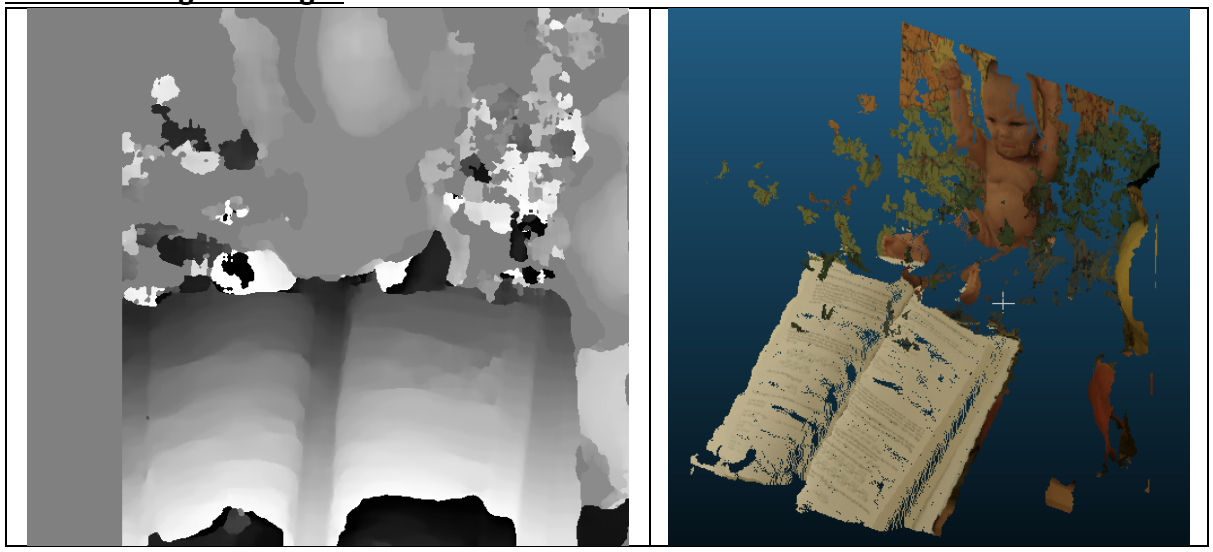
1. minDisparity (38): Minimum possible disparity value.
2. numDisparities (60): Maximum disparity minus minimum disparity
3. blockSize (30): Matched block size.
4. P1 (60): The first parameter controlling the disparity smoothness.
5. P2 (100): The second parameter controlling the disparity smoothness. The larger the values are, the smoother the disparity is. P1 is the penalty on the disparity change by plus or minus 1 between neighbor pixels. P2 is the penalty on the disparity change by more than 1 between neighbor pixels.
6. speckleRange (6): Maximum disparity variation within each connected component.
7. speckleWindowSize (250): Maximum size of smooth disparity regions to consider their noise speckles and invalidate.
8. disp12MaxDiff (30): Maximum allowed difference (in integer pixel units) in the left-right disparity check.

These values provided a good disparity map.

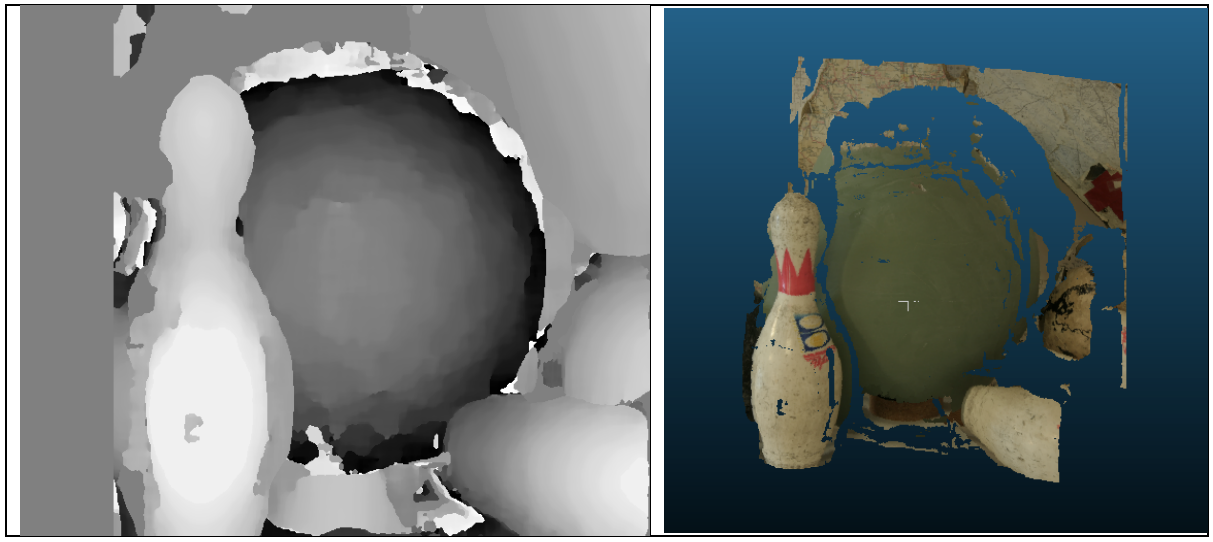
Once the disparity map was calculated and transformed to depth information using the conversion coefficient, a data matrix containing the information for the .ply file was created in the required format. A .ply file was created using the file writing operations in Python. A header was written first with the required information and then the data was appended to the header file.

Trials to create a pointcloud from an image of our choosing was difficult. Computation complexity was the first major difficulty due to the high resolution of the images captured by the camera. When the images were reduced in resolution, the result was only slightly better. This can be improved upon by changing the parametrs in the SGBM class and this change is attributed to the different camera and object placement in the images.

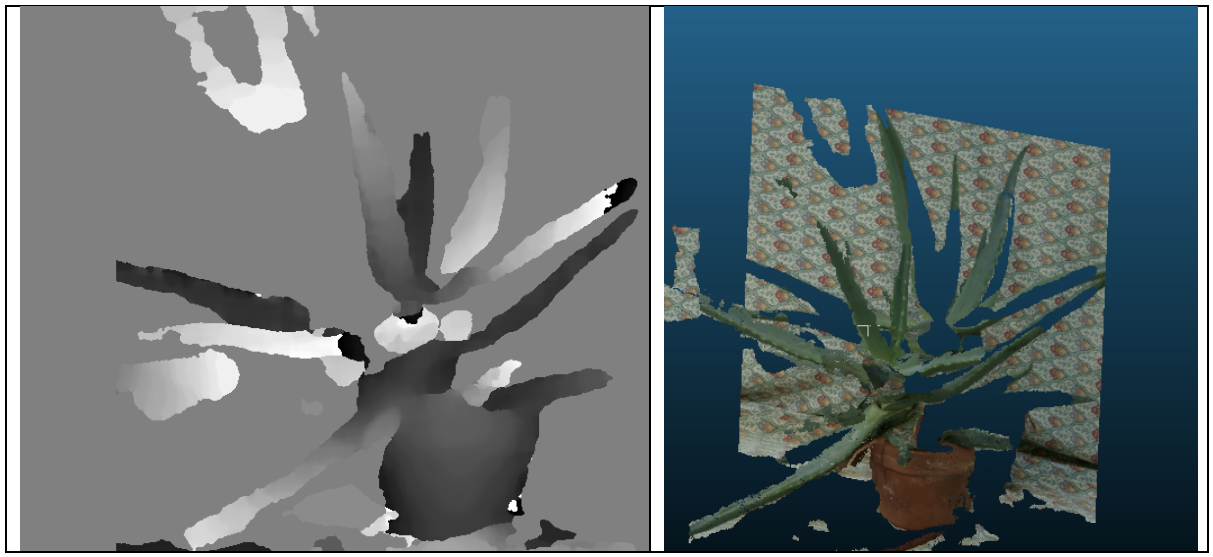
Results of original images



Baby image results

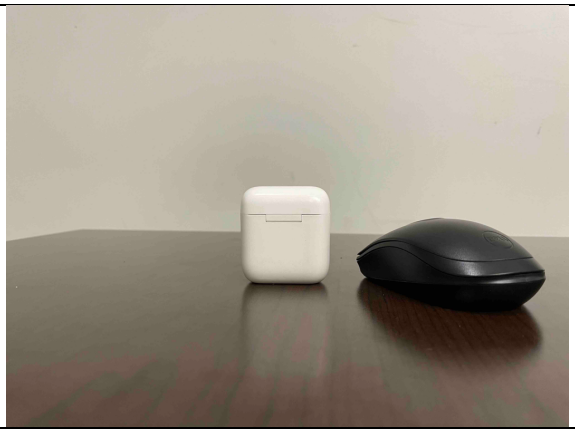


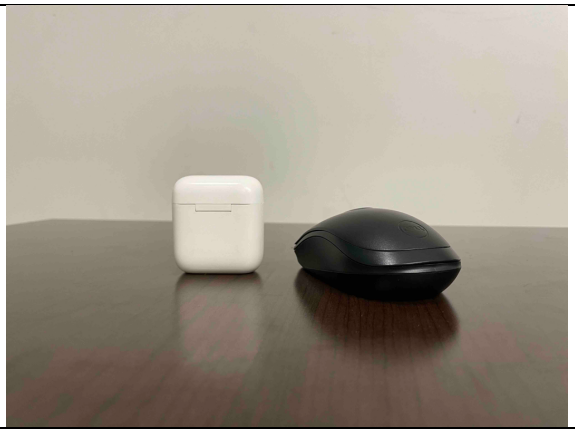
Ball image results

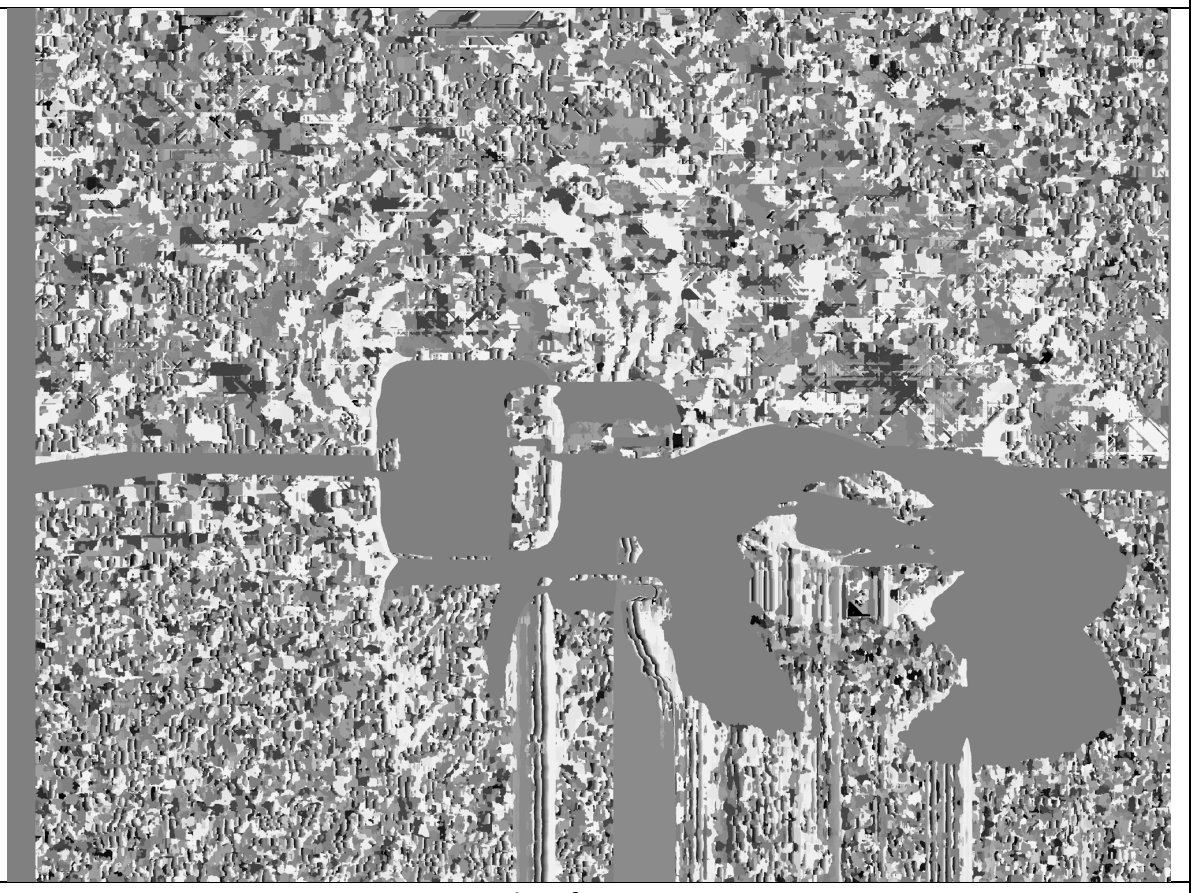


Plant Image results

Results of New images







Results of new image

Source Code

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

def createPlyFile(data,file):
    # Opening the file
    f = open(f"ps7-ply/{file}.ply","w")
    # Creating the header
    f.write("ply\n")
    f.write("format ascii 1.0\n")
    f.write(f"element vertex {data.shape[0]}\n")
    f.write(f"property float32 x\n")
    f.write(f"property float32 y\n")
    f.write(f"property float32 z\n")
    f.write(f"property uint8 red\n")
    f.write(f"property uint8 green\n")
    f.write(f"property uint8 blue\n")
    f.write("end_header\n")
    # Adding the data
    for i in data:
        f.write(f"{i[0]} {i[1]} {i[2]} {i[3]} {i[4]} {i[5]}\n")
    # CLose file
    f.close()

file = input("Enter file id : ")
img_L = cv2.imread("ps7-images/"+file+"-left.png")
img_l = cv2.cvtColor(img_L, cv2.COLOR_BGR2GRAY)

img_R = cv2.imread("ps7-images/"+file+"-right.png")
img_r = cv2.cvtColor(img_R, cv2.COLOR_BGR2GRAY)

# Creating an object of StereoBM algorithm
'''In block matching or cv2.StereoBM_create() the disparity is computed
by comparing the sum of absolute
differences (SAD) of each 'block' of pixels. In semi-global block
matching or cv2.StereoSGBM_create()
forces similar disparity on neighbouring blocks. This creates a more
complete disparity map but is more
computationally expensive.
'''

stereo = cv2.StereoSGBM_create(minDisparity=38, numDisparities=60,
blockSize = 30, P1=60, P2=100,
```

```
 speckleRange=6, speckleWindowSize=250, disp12MaxDiff=30)
# stereo = cv2.StereoSGBM_create(minDisparity=40, numDisparities=60,
blockSize = 12)
k = 0.65
disparity = k*stereo.compute(img_l,img_r)

cv2.imshow('Disparity Map', np.uint8(disparity))
cv2.waitKey(0)
cv2.imwrite("ps7-images/"+file+"-disparity.png",np.uint8(disparity))
'''
# SPLIT INTO THREE CHANNELS. X AND Y POSITION OF ANY PIXEL. KxDisparity
IS Z. RGB FROM THREE CHANNELS
'''
(B,G,R) = cv2.split(img_L)
data = np.array([[i,j,int(disparity[i,j]),R[i,j],G[i,j],B[i,j]] for j
in range(38+60,img_L.shape[1]) for i in range(img_L.shape[0])])

createPlyFile(data, file)

cv2.destroyAllWindows()
```

System Specifications

Operating System: macOS Monterey Version 12.5.1
Hardware: MacBook Air 2017 (Intel Core i5)
Python: Conda environment utilizing Python 3.9.1
IDE: Visual Studio Code

Time taken: Q1: 4 hour