CSE-483: Mobile Robotics **Monsoon 2019**

Assignment - 3

Prepared by: AadilMehdi Sanchawala, Rahul Sajnani

In this report, we discuss our approach to doing the assignment questions. Instructions to run the assignment are present in the end of this report.

1 Point Cloud Generation - (q1_script.py)

1.1 Stereo Normal Assumption

In the case of stereo-normal images we make the following assumptions

1. Image planes are parallel and not rotated with respect to each other
2. Image offset is only in the x direction, we call this disparity or parallax in x direction

$$p_x = c_x^1 - c_x^2 = d_x \tag{1}$$

3. Disparity in y direction is zero for all corresponding points of the stereo-normal pairs of images.

$$p_y = c_y^1 - c_y^2 = 0 \tag{2}$$

1.2 Stereo Normal Triangulation

For stereo-normal triangulation, we can calculate the X, Y and Z co-ordinates of the 3D world point using the following

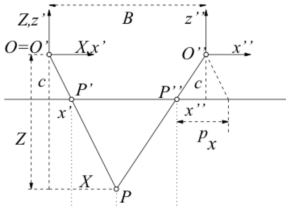

Figure 1: Stereo-Normal Triangulation

$$Z = c \cdot \frac{B}{x_\prime - x_{\prime\prime}} = c \cdot \frac{B}{p_x} \tag{3}$$

$$X = x_\prime \cdot \frac{B}{x_\prime - x_{\prime\prime}} = x_\prime \cdot \frac{B}{p_x} \tag{4}$$

$$Y = y_\prime \cdot \frac{B}{x_\prime - x_{\prime\prime}} = y_\prime \cdot \frac{B}{p_x} \tag{5}$$

where B is the Baseline, i.e. the distance between the stereo-cameras, $x_\prime$ and $x_{\prime\prime}$ are the x-coordinates of the image pixels of the left and right stereo-image pairs respectively,c is the camera constant or focal length, and p_x is the disparity in x.

1.3 Parallax Map

Using Homogeneous Co-ordinates we can write the equations in the Section 1.2 in a very simple form

$$\begin{bmatrix} U \\ V \\ W \\ T \end{bmatrix} = \begin{bmatrix} B & 0 & 0 & -c_x \\ 0 & B & 0 & -c_y \\ 0 & 0 & B.c & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} x_{'} \\ y_{'} \\ 1 \\ p_x \end{bmatrix} \tag{6}$$

OR

$$\begin{bmatrix} U \\ V \\ W \\ T \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 0 & c \\ 0 & 0 & -1/B & 0 \end{bmatrix} \cdot \begin{bmatrix} x_{'} \\ y_{'} \\ p_x \\ 1 \end{bmatrix} \tag{7}$$

where B is the Baselength, c_x is the sensor shift in x, c_y is the sensor shift in y, c is the camera constant, $x_{'}$ is the x co-ordinate in left image, $y_{'}$ is the y image co-ordinate in right image, and p_x is the disparity.

1.4 Computing the Parallax Map

We computed the Parallax Map for every pair of stereo normal images using the Semi Global Block Matching Algorithm. We used the following parameters for the SGBM algorithm,

```
window_size = 5
minDisparity = -39
numDisparities=144
stereo = cv2.StereoSGBM_create(minDisparity=-39,
    numDisparities=144,
    blockSize=5,
    P1=8 * 3 * window_size ** 2,
    P2=64 * 3 * window_size ** 2,
    disp12MaxDiff=1,
    uniquenessRatio=10,
    speckleWindowSize=100,
    speckleRange=32,
    preFilterCap=63
)
```

1.5 Point Cloud Construction Registration

1. We have computed the point cloud using the above mentioned disparity maps and triangulation.
2. We construct the point cloud for each of the given stereo-normal images.
3. Then we use the pose of given, to find the absolute orientation, with respect to the world origin and registered all the local point clouds into a global point cloud, of the entire scene.

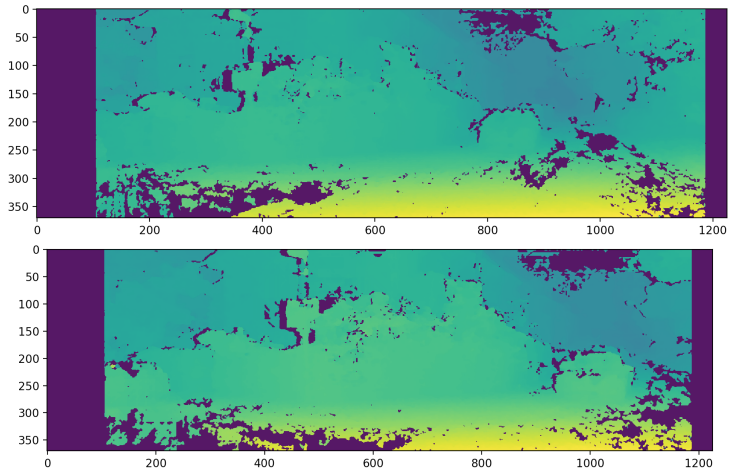

Figure 2: Disparity Images

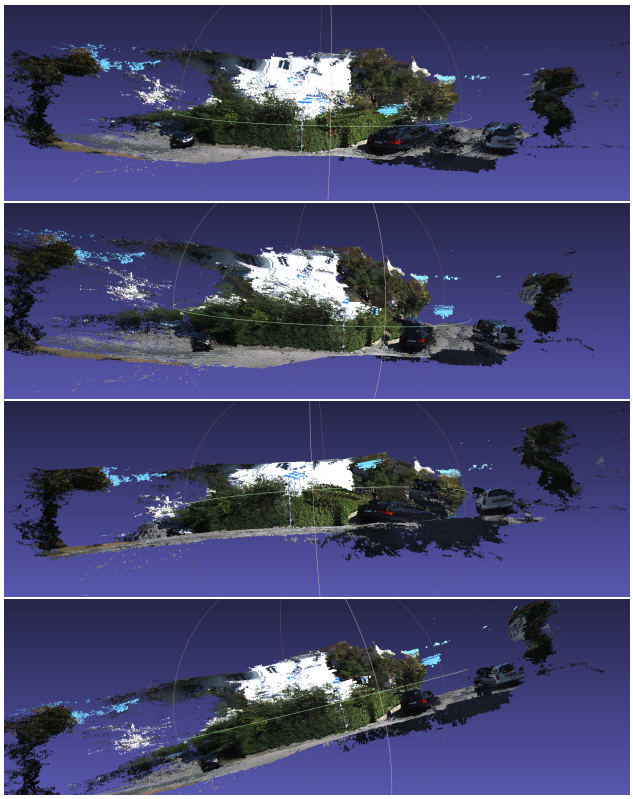

Figure 3: Constructed Point Cloud

1.6 Bonus - (q1_bonus_script.py)

We generated 2 different point clouds. One in STEREO_SGBM_MODE_3WAY and another in HH mode. The 3WAY SGBM mode is very similar to the default SGBM output. But HH mode output has missing sky and improper object edges (Cars in our case).

Figure 4: Imaging the Point Cloud

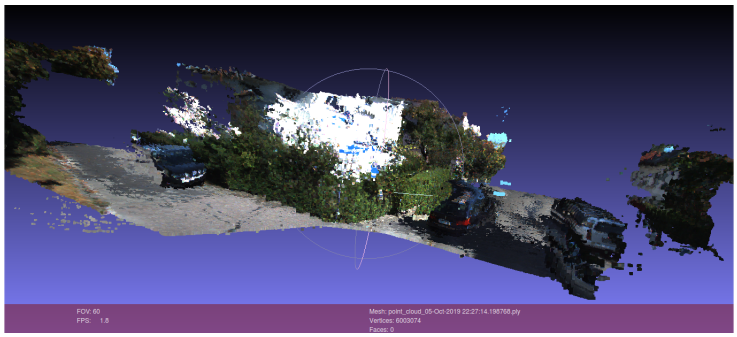

Figure 5: HH mode point cloud output

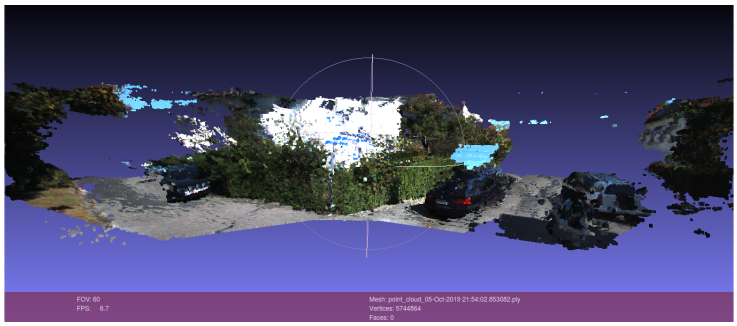

Figure 6: STEREO_SGBM_3WAY mode point cloud output

2 Motion estimation using iterative PnP - (q2_script.py)

- We first take any one pair of the stereo images provided and create an individual point cloud and then project this point cloud to the world frame by multiplying it with poses of that particular frame.
- Once we have the point cloud in world frame and we know its corresponding image coordinates we do PnP to estimate its initial pose which will be our initialization for Gauss Newton minimization. x is our homogeneous image coordinates and X is our homogeneous points in world frame. R_w^c is rotation from world to camera frame. T_w^c is translation from world to camera frame.

$$x = K[R_w^c|T_w^c]X$$
$$AP = 0$$
Where, A =

$$\begin{bmatrix} -X_{11} & -X_{12} & -X_{13} & -1 & 0 & 0 & 0 & 0 & x_{11}X_{11} & x_{11}X_{12} \\ x_{11}X_{13} & x_{11} & & & & & & & & \\ 0 & 0 & 0 & 0 & -X_{11} & -X_{12} & -X_{13} & -1 & x_{12}X_{11} & x_{12}X_{12} \\ x_{12}X_{13} & x_{12} & & & & & & & & \\ \dots & & & & & & & & & \end{bmatrix}$$

U, D, Vt = SVD(A_{Nx12})
We get P = reshape(Vt[:,-1],(3,4))
R = P[:3,:3]
Ur, Dr, Vrt = SVD(R)
R = Ur · Vrt
T = P[:3,3] / σ_1
Here σ_1 is the first eigenvalue of Dr

- Now we initialize our calculated R and T matrices as the initial value of Gauss Newton minimization. We calculate the jacobian J.
- We know $E = J\Delta P$. We get $J'E = J'J\Delta P$. Now our update $\Delta P = (J'J)^{-1}J'E$. We iterate and break out of the loop if the mean squared error change between current and previous error is less than 0.0000001.
- After recovering the pose from Gauss Newton minimization we reconstruct the images to from the ground truth vs our estimated pose and compare them.
  Calculated $[R|T]$ from camera to world:

$$\begin{bmatrix} -9.09854769e-01 & 5.44537787e-02 & -4.11338164e-01 & -1.87283491e+02 \\ 4.11782635e-02 & 9.98307202e-01 & 4.10740946e-02 & 1.87021461e+00 \\ 4.12878426e-01 & 2.04332716e-02 & -9.10556910e-01 & 5.41708340e+01 \end{bmatrix}$$

  Actual $[R|T]$ from camera to world:

$$\begin{bmatrix} -9.098548e-01 & 5.445376e-02 & -4.113381e-01 & -1.872835e+02 \\ 4.117828e-02 & 9.983072e-01 & 4.107410e-02 & 1.870218e+00 \\ 4.128785e-01 & 2.043327e-02 & -9.105569e-01 & 5.417085e+01 \end{bmatrix}$$

- The above calculated pose is for the first frame. Our calculated value is very close to the ground truth. This also confirms that our point cloud is in the appropriate metric scale.

2.1 Outputs

Figure 7: Reconstructed image with ground truth pose for image 5.

Figure 8: Reconstructed image (our calculated pose) for image 5

Figure 9: Reconstructed image with ground truth pose for image 0.

Figure 10: Reconstructed image (our calculated pose) for image 0

3 Instructions

Run the following instructions in order to execute the script.

```
pip install -r requirements.txt

python3 q1_script.py
// Output stored in output_question_1 folder

python3 q2_script.py
// Output stored in output_question_2 folder

python q1_bonus_script.py
// Output stored in output_question_1_bonus folder
```

4 Contribution

Both teammates have contributed equally in both the questions. Both the teammates have researched on the topics, required for completing the assignment. Different parts of the assignment were coded and debugged equally by each of the teammates.