ENPM673

Perception for Autonomous Robots — PROJECT-4

26 April 2023

Vamshi Kalavagunta

119126332

vamshik@umd.edu

A. James Clark School of Engineering

University of Maryland College Park - 20742

1 PROJECT DESCRIPTION:

In this project, we are going to implement the concept of Stereo Vision. We will be given 3 different datasets, each of them containing 2 images of the same scenario but taken from two different camera angles. By comparing the information about a scene from 2 vantage points, we can obtain the 3D information by examining the relative positions of objects.

1.1 Calibration:

Calibration aspect of this project was to compute fundamental and essential matrices given the correspondence points.

Image processing

Image processing is a fundamental aspect of computer vision, which is widely used in various fields such as robotics, autonomous vehicles, and image recognition. In this context, one common task is to match images, i.e., find correspondences between points in different images. In this process, it is often useful to extract and match key features, which are local regions of the image that can be identified robustly and efficiently. The Scale-Invariant Feature Transform (SIFT) algorithm is a popular method for detecting such key features.

The following are the steps involved in matching images using SIFT and Brute-Force Matcher:

1. **Read the images from the dataset:** In order to match two images, we need to load them into memory. We can use OpenCV's imread() function to read the images in various formats such as JPG, PNG, and BMP.

2. **Convert the images to Grayscale and RGB:** Before applying any image processing algorithm, it is often necessary to convert the images to a suitable format. Grayscale images are simpler to process and require less memory than RGB images, which consist of three color channels. However, RGB images can be more informative as they contain color information that can aid in feature detection and matching.

3. **Extract key features using SIFT:** Once we have loaded the images, we can extract the key features using SIFT. SIFT detects the local features in an image that are invariant to scale and rotation, and returns a set of keypoints and descriptors that characterize the detected features. We can use OpenCV's SIFT-create() function to create a SIFT object, and detectAndCompute() method to obtain the keypoints and descriptors for both images.

4. **Match the key features using Brute-Force Matcher:** The next step is to match the keypoints from both images using Brute-Force Matcher. Brute-Force Matcher compares each descriptor of the first image with every descriptor of the second image and returns the best matches. We can use OpenCV's BFMatcher() function to create a Brute-Force Matcher object, and match() method to obtain the matches between the descriptors of both images.

5. **Sort the matched points based on distance:** The matches obtained in the previous step may include some false matches. In order to filter out these false matches, we can sort the matches based on the distance between their descriptors. Matches with lower distances between their descriptors are more likely to be true correspondences. We can use Python's sorted() function to sort the matches based on distance.

6. **Extract the correspondences:** Once the matches are sorted, we can extract the correspondences between the keypoints of both images. Correspondences are pairs of keypoints from both images that are spatially related, i.e., they represent the same point in the scene. We can use the sorted matches to extract the correspondences and obtain a list of pairs of points representing the correspondences. These correspondences can be used for further processing, such as computing the fundamental matrix and stereo rectification.

7. **Fundamental matrix:** The fundamental matrix provides a relationship between two images of the same scene taken from different cameras or the same camera placed at different points. It describes the constraints on the projection of points from the scene in both images. The F matrix can be estimated from feature correspondences obtained from feature matching. It is an algebraic representation of the epipolar geometry.

8. **Hartley algorithm:** we can use the 8-point Hartley algorithm to calculate the fundamental matrix. This algorithm is commonly used in computer vision to estimate the fundamental and essential matrices for stereo vision from matching sets of corresponding points in multiple images.

$$\begin{bmatrix} u_1u_1' & u_1v_1' & u_1 & v_1u_1' & v_1v_1' & v_1 & u_1' & v_1' & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_nu_v' & u_nv_n' & u_n & v_nu_n' & v_nv_n' & v_n & u_n' & v_n' & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ \vdots \\ f_{33} \end{bmatrix} = 0$$

9. **Normalization:** In this step, the values of X and X′ were normalized, where X = $[u, v, 1]$ and X′ = $[u', v', 1]$ are homogeneous point correspondences between the images. The fundamental matrix F was obtained by solving the equation A.f = 0, where f is a 9×1 vector that was computed using SVD and reshaped into a 3×3 matrix.

10. **Epipoles:** To perform stereo vision for images, it is important to have a F matrix with a rank of 2. However, in many cases, due to noise in the point correspondences of the images in the dataset, the estimated F matrix may have a rank of 3. In order to ensure that the fundamental matrix always has a rank of 2, we constrain it by setting the last singular value of the estimated F to zero. If the fundamental matrix has a full rank, it will have a null space, which means that epipoles will not exist for image planes.

11. **Corresponding points:** To obtain the 8 best point correspondences from the sorted matches and compute the best fundamental matrix possible, we use the RANSAC algorithm. This algorithm removes the noise we get from all the other feature descriptors when we use SIFT to get matching features, giving us the best point correspondences. The goal is to minimize the terms of the Homogeneous equation.

12. **Essential Matrix:** The Essential Matrix is a crucial mathematical concept that encodes epipolar geometry. It is a 3×3 matrix that provides valuable information about the relative positions and orientations of the two cameras used to capture the two views of a scene. Specifically, if we have a point in one image, multiplying it by the Essential Matrix will provide us with the epipolar line in the second view.

    - To compute the Essential Matrix, we require the intrinsic parameters of the cameras, which are stored in the calib.txt file as K1 and K2. Using these parameters, we can estimate the relative camera poses between the two views by computing the Essential Matrix as follows:

    $$E = K_2^T F K_1$$

    - where F is the Fundamental Matrix obtained from the corresponding points in the two images. The Essential Matrix is a useful tool in computer vision as it allows us to rectify images, which simplifies the process of stereo matching and depth estimation.

13. **Recovering the Camera Pose :** In computer vision and 3D reconstruction, the camera pose represents the position and orientation of the camera with respect to the world coordinate system. A camera pose can be defined by 6 degrees of freedom: three angles of rotation (roll, pitch, and yaw) and three components of translation (x, y, and z).

- To estimate the camera pose, we can use the correspondences between points on the image plane and 3D points in the world. The essential matrix E is a fundamental matrix that describes the relationship between these correspondences. It can be decomposed into four possible rotation and translation matrices.
- To recover the pose of the second camera with respect to the first camera, we assume that the first camera is located at the origin of the world coordinate system. Once we retrieve the four possible configurations of rotation and translation matrices, we need to find the unique camera pose that satisfies the given triangulated points.
- To achieve this, we check the chirality condition. This condition states that a point that lies in an image must be in front of the camera that produced that image. Therefore, the reconstructed 3D points must be in front of the cameras. To check the cheirality condition, we triangulate the 3D points given the two camera poses using linear least squares and check the sign of the depth in the camera coordinate system with respect to the camera center.
- The best camera configuration is the one that produces the maximum number of points satisfying the chirality condition. By imposing the chirality condition, we can distinguish between the points in front of the camera and the ones behind it, and thus find the correct camera pose.

1.2 Rectification:

Rectification is a process that aligns an image to its axis. It transforms the pairs of conjugate epipolar lines in the images so that they become collinear and parallel to the horizontal axis, which is known as the baseline of the image.

1. To achieve rectification, we perform image warping, or perspective transformation, to align the input images so that their epipolar lines become horizontal and pass through the same point in the image. This makes it easier to search for corresponding points in rectified images.

2. We use the fundamental matrix and feature points obtained from images through SIFT to construct epipolar lines for both images. Epipolar lines usually intersect at the optic center of the image, and the point of intersection shifts as the camera moves. However, for further computation, we need to make the epipolar lines parallel to each other. This can be done by reprojecting image planes onto a common plane parallel to the line between the camera centers.

3. To perform rectification, we use the cv2.stereorectifyUncalibrated() function, which applies homographic transformation to project the image plane of each camera onto a perfectly aligned virtual plane. This transformation is computed from a set of matched point correspondences, matched images, and a fundamental matrix.

4. After rectification, the epipolar lines become parallel. The next step is to transform the feature points using the obtained homography matrices. These feature points constitute the epipolar points through which pass out epipolar lines. When the images are rectified, the F matrix and epipolar geometry are modified.

1.3 Correspondence:

In the correspondence step of stereo vision, we aim to find the corresponding matching regions in each image. We use the matching features obtained in previous steps to determine these regions.

To find a corresponding match for every point in image 1, we search along the epipolar line in image 2 using epipolar geometry. This allows us to limit our search area to a line, rather than the entire image. We employ block matching to achieve this. Specifically, we slide a window of a predetermined size over each point in image 1 and search for the closest corresponding match in image 2.

1. We use the Sum of Absolute Differences (SAD) approach for block matching. Initially, we tried using the same window size for all images in the dataset, but the results were unsatisfactory. Therefore, we tuned the window size for each dataset, which resulted in better disparity heat maps.

2. The SAD is given by the formula:

$$s = \sum_{(u,v)\in \mathbf{I}} |\mathbf{I}_1[u,v] - \mathbf{I}_2[u,v]|$$

3. To perform block matching, we select a window size and slide it across the epipolar line along every matching pixel coordinate index in both images. We then compute the SAD between both the windowed regions and select the corresponding match with the least SAD value. The matched values in both images correspond to the same point in the world scene, viewed from different perspectives. Using this approach, we compute disparity for all windowed regions along the corresponding epipolar lines from both images.

4. The SAD approach is time-consuming and depends on the homogeneity of the images used for block matching. A smaller window size results in blocks with smaller variance values, which require fewer computations for block searching.

1.4 Depth Calculation:

Depth calculation is a crucial step in stereo vision. In order to perform depth calculation, we first need to have the camera and image parameters. These parameters are usually provided in the calib.txt file.

1. Once we have the camera and image parameters, we can find the disparity between the images. The depth can then be calculated using the following formula:

$$depth = \frac{focallength * baseline}{disparity}$$

2. However, when I initially tried to calculate the depth for each image set, I found that the results were not accurate. To improve the accuracy, I tunned the window size and thresholding parameter to filter out absurd depth values.

3.

2 Results

2.1 Reference

1. https://cmsc733.github.io/2022/proj/p3/fundmatrix

```
Found 2365 matches and selected 100 of the best ones

The Estimated Fundamental Matrix:
 [[ 9.49286911e-12  9.96820444e-08 -2.72284616e-05]
 [-9.56622535e-08  6.76165036e-10  3.90242093e-03]
 [ 2.57161787e-05 -3.89620434e-03  1.55023918e-05]]

The Estimated Essential Matrix:
 [[ 2.93459967e-05  3.08154354e-01  4.90223533e-02]
 [-2.95727682e-01  2.09027815e-03  6.72255081e+00]
 [-4.77467239e-02 -6.70444606e+00  4.25356111e-03]]

 Rotation matrix
 [[ 9.95858934e-01 -1.43634188e-02  8.97701308e-02]
 [-1.44060269e-02 -9.99896213e-01 -1.73302694e-04]
 [ 8.97633030e-02 -1.12064589e-03 -9.95962496e-01]]

 Translation matrix
 [-0.99891878  0.00731339 -0.04591071]

Estimated Homography Matrix H1:
 [[ 3.69317880e-03 -2.91271334e-04 -2.01028197e-01]
 [-2.63717274e-05  3.89659494e-03  2.66228513e-02]
 [-9.80685920e-08  1.40964726e-09  4.00039155e-03]]

Estimated Homography Matrix H2:
 [[ 9.74795983e-01 -6.83450503e-03  2.78864894e+01]
 [-7.15239131e-03  1.00007473e+00  6.82594403e+00]
 [-2.62285830e-05  1.83894256e-07  1.02508014e+00]]
```

Figure 1: Chess Dataset.

```
Found 5021 matches and selected 100 of the best ones

The Estimated Fundamental Matrix:
 [[-1.65479995e-10 -3.34640559e-08 -1.03115514e-04]
 [ 3.68388311e-08 -2.57240161e-09  2.45010402e-03]
 [ 1.01052891e-04 -2.45007979e-03 -1.18893104e-03]]

The Estimated Essential Matrix:
 [[-4.97649794e-04 -1.00636820e-01 -2.34512085e-01]
 [ 1.10785818e-01 -7.73601141e-03  4.26590339e+00]
 [ 2.36350656e-01 -4.27245730e+00 -3.15485371e-03]]

 Rotation matrix
 [[ 0.99271798  0.10992461 -0.04927055]
 [ 0.10985879 -0.99393895 -0.0040501 ]
 [-0.04941713 -0.0013922  -0.99877726]]

 Translation matrix
 [-0.99821494 -0.05485801  0.02361207]

Estimated Homography Matrix H1:
 [[-2.48919846e-03 -9.46583356e-05  2.23900940e-01]
 [ 1.00204197e-04 -2.45026082e-03 -5.82956966e-02]
 [-3.64999008e-08  7.30659054e-10 -2.43256471e-03]]

Estimated Homography Matrix H2:
 [[ 1.00582332e+00  5.51475492e-02 -5.60862383e+01]
 [-4.17273645e-02  9.99214102e-01  2.32872385e+01]
 [ 1.35611218e-05  7.43532808e-07  9.91963203e-01]]
```

Figure 2: Ladder Dataset.

```
Found 3532 matches and selected 100 of the best ones

The Estimated Fundamental Matrix:
 [[ 1.23649204e-11  2.19309379e-08  2.36383143e-05]
 [-2.31419938e-08  9.35459016e-09 -2.43980315e-03]
 [-2.34672303e-05  2.43642498e-03 -3.03356738e-03]]

The Estimated Essential Matrix:
 [[ 3.71671501e-05  6.59212058e-02  6.16037118e-02]
 [-6.95614637e-02  2.81185359e-02 -4.25298332e+00]
 [-6.24109106e-02  4.26304017e+00 -2.49386452e-03]]

 Rotation matrix
 [[ 9.99999720e-01  8.96781061e-05 -7.43566808e-04]
 [-8.74440829e-05  9.99995484e-01  3.00395706e-03]
 [ 7.43832839e-04 -3.00389120e-03  9.99995212e-01]]

 Translation matrix
 [-0.999774   -0.01449067  0.01555551]

Estimated Homography Matrix H1:
 [[ 2.47437678e-03 -4.99066636e-05 -2.08689608e-01]
 [-2.32534522e-05  2.43636893e-03  1.95354454e-02]
 [ 2.29393819e-08 -8.95837884e-09  2.41876229e-03]]

Estimated Homography Matrix H2:
 [[ 1.00849608e+00  1.45780394e-02 -1.60283775e+01]
 [-9.61591303e-03  9.99965471e-01  9.24992210e+00]
 [ 8.95889566e-06  1.29502868e-07  9.91329529e-01]]
```

Figure 3: Artroom Dataset.

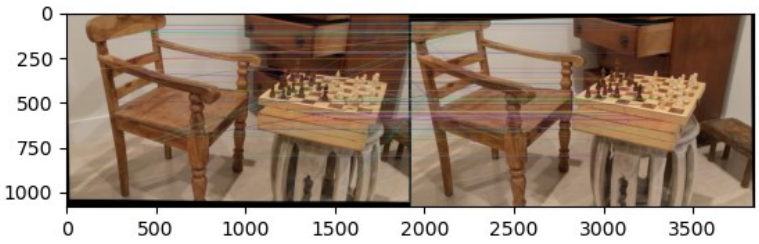

Figure 4: Chess matched image.

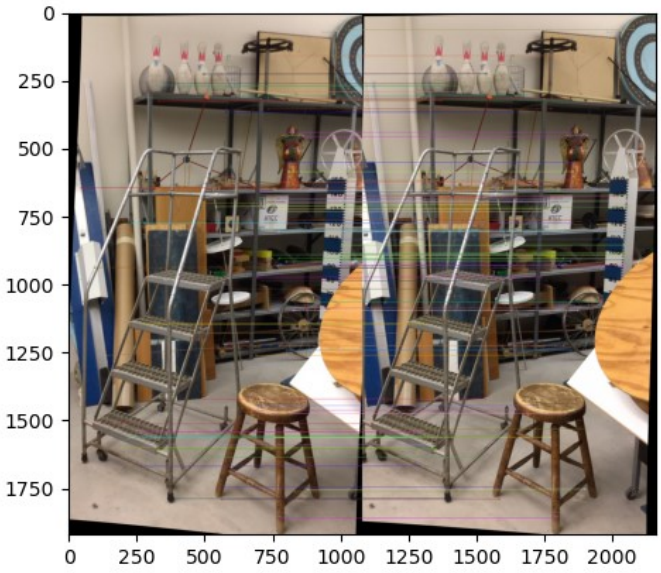

Figure 5: Ladder matched image.

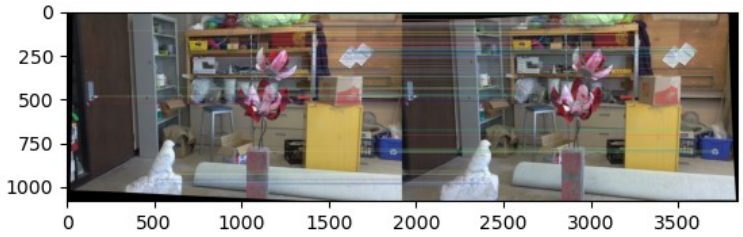

Figure 6: Artroom matched image.

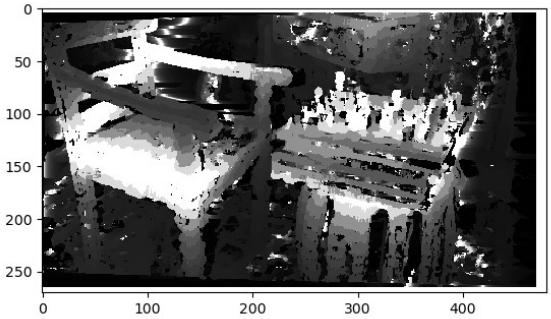

Figure 7: Chess Depth image- gray.

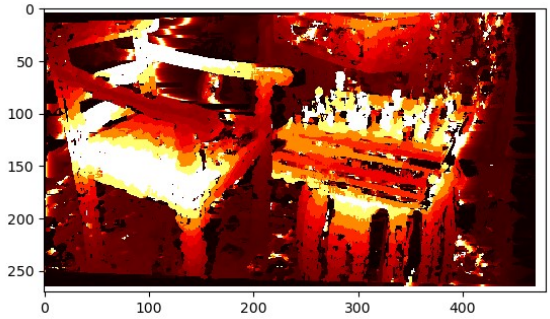

Figure 8: ChessDepth image- Heat.

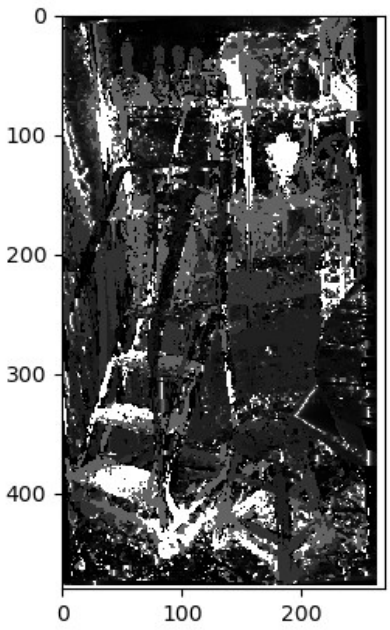

Figure 9: Ladder Depth image- gray.

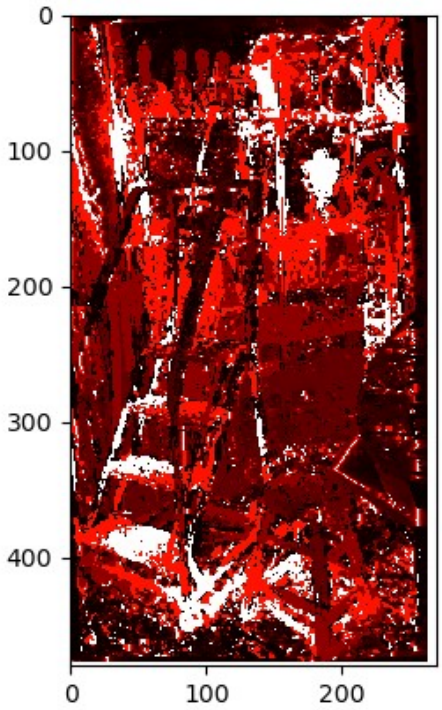

Figure 10: Ladder Depth image- Heat.

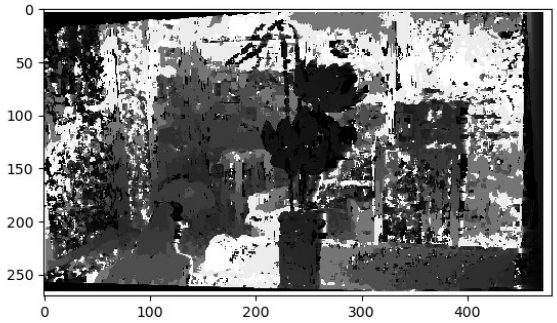

Figure 11: Artroom Depth image- gray.

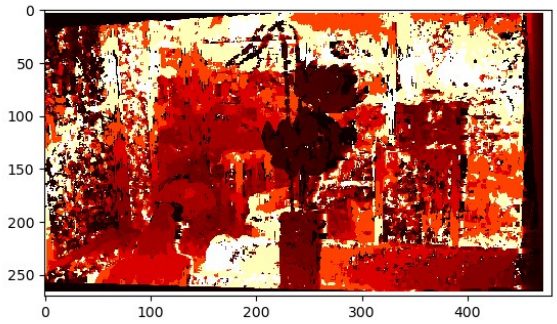

Figure 12: Artroom Depth image- Heat.

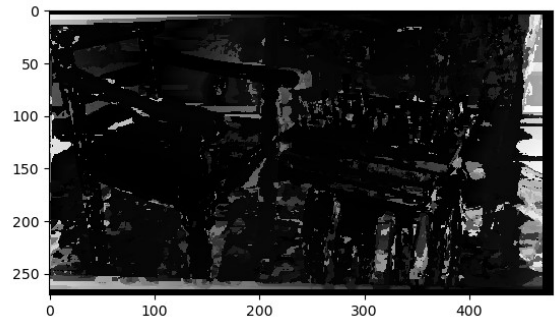

Figure 13: Chess Disparity image- gray.

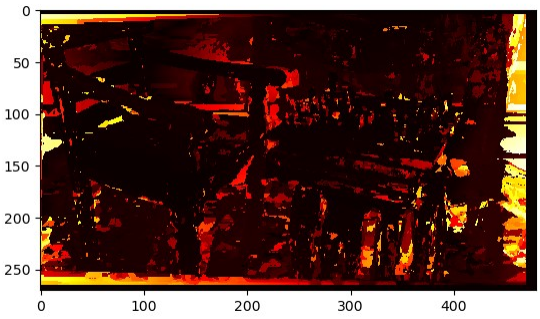

Figure 14: Chess Disparity image- Heat.

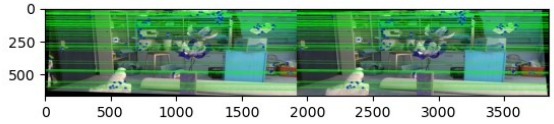

Figure 15: Artroom Epipolar lines.

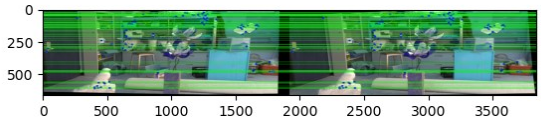

Figure 16: Artroom Rectified Epipolar lines.