ENPM-673 PERCEPTION FOR AUTONOMOUS ROBOTS

# Project-3
# Creating a Stereo Vision System

Jeet Patel (117430479)

# Contents

# 1. Introduction

The principle of stereo vision is implemented in this project. We are provided with three datasets, each of which includes two photographs of the same situation shot from two separate camera angles. We may gain 3D information by comparing information about a scene from two vantage points and observing the relative locations of objects.

## 1.1 Stereo Vision

Stereo vision is the process of recovering depth from camera images by comparing two or more views of the same scene. Simple, binocular stereo uses only two images, typically taken with parallel cameras that were separated by a horizontal distance known as the "baseline." The output of the stereo computation is a disparity map (which is translatable to a range image) which tells how far each point in the physical scene was from the camera.

## 1.2 Overview of pipeline

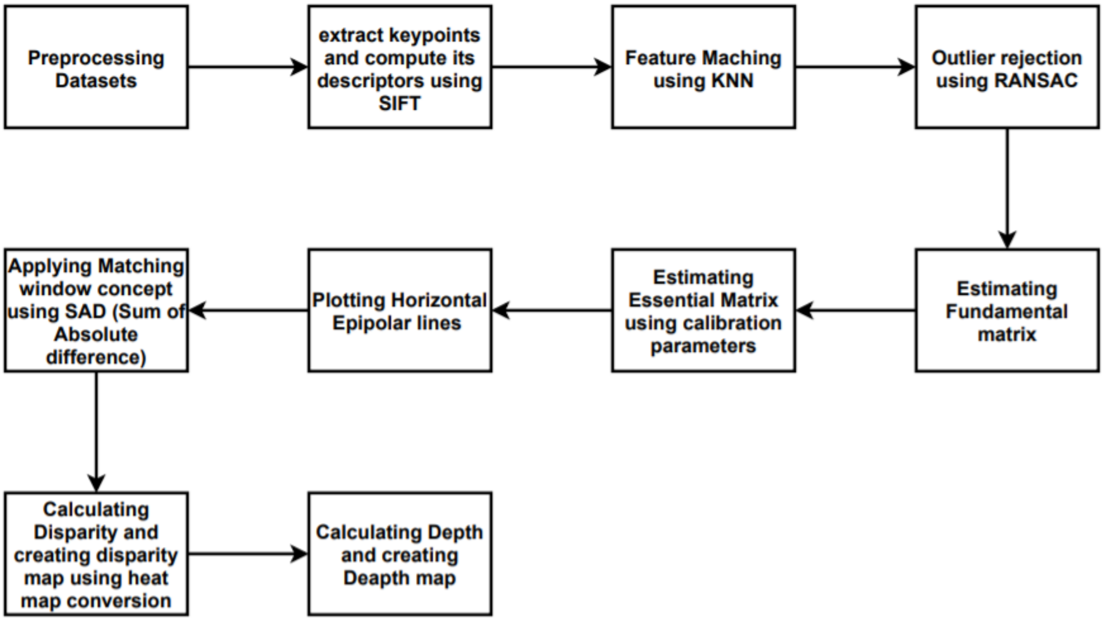

Preprocessing Datasets
extract keypoints and compute its descriptors using SIFT
Feature Maching using KNN
Outlier rejection using RANSAC
Estimating Fundamental matrix
Estimating Essential Matrix using calibration parameters
Plotting Horizontal Epipolar lines
Applying Matching window concept using SAD (Sum of Absolute difference)
Calculating Disparity and creating disparity map using heat map conversion
Calculating Depth and creating Deapth map

Figure-1: Pipeline for Creating stereo vision system

## 2. Calibration

The task here is to compare the two images and selecting the set of matching features. This is accomplished using SIFT.

### 2.1 Feature extraction and matching

- Matching features were extracted for consecutive frames using SIFT feature descriptor.
- SIFT (Scale-Invariant Feature Transform) is a method for extracting feature vectors that describe local patches of an image. Not only are these feature vectors scale-invariant, but they are also invariant to translation, rotation, and illumination.
- Here, SIFT give out the key points and its descriptors.
- Following the identification of features in both frames, we could use a BFMatcher.knnMatch() Matcher to align the features that are identical. After that, applying the ratio test to get best matches.
- This key point will be useful for estimating fundamental matrix.

### 2.2 Estimating Fundamental Matrix

- The Fundamental matrix can be computed by choosing eight points at random from the previously extracted points.
- The Fundamental Matrix, abbreviated as F, is a three-dimensional matrix of rank two. It connects the corresponding points in two pictures.
- It is the algebraic representation of epipolar geometry. It only depends on the cameras' internal parameters (K matrix) and the relative pose i.e., it is independent of the scene structure.
- Since, F is a 3×3 matrix, we can set up a homogenous linear system with 9 unknowns:

$$\begin{bmatrix} x_i' & y_i' & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = 0$$

$$x_i x_i' f_{11} + x_i y_i' f_{21} + x_i f_{31} + y_i x_i' f_{12} + y_i y_i' f_{22} + y_i f_{32} + x_i' f_{13} + y_i' f_{23} + f_{33} = 0$$

- Here, the above equation has nine unknowns. Simplifying the equation for m correspondences we get,

$$\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y1 & x'_1 & y'_1 & 1 \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ x_mx'_m & x_my'_m & x_m & y_mx'_m & y_my'_m & y_m & x'_m & y'_m & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{21} \\ f_{31} \\ f12 \\ f_{22} \\ f_{32} \\ f_{13} \\ f_{23} \\ f_{33} \end{bmatrix} = 0$$

- This system of equations can be solved using Singular Value Decomposition to solve the linear least squares.
- We can approximate the Fundamental Matrix by taking 8 random correspondences.

### 2.3 Matching Outliers using RANSAC

- The data is bound to be noisy and (in general) includes many outliers.
- Thus, to remove these outliers, RANSAC outlier’s rejection technique is implemented to obtain the best fundamental matrix.
- Out of numerous possibilities, best pair with minimum number of inliers is chosen to estimate the fundamental matrix.
- If the distance is greater than that of the threshold set, we discard the key point. Here we have used threshold to be 0.5.
- We consider the key point to be a reasonable fit if its distance from the epipolar line is less than the threshold value, and we account it as the inlier.

### 2.4 Estimating Essential Matrix

- Estimation matrix determines camera poses between the two images.
- Essential matrix is another 3×3 matrix, but with some additional properties that relates the corresponding points if the cameras obey the pinhole model.
- It is derived from camera calibration matrix and fundamental matrix.

$$E = K^T F K$$

- As in the case of F matrix computation, the singular values of E are not necessarily (1,1,0) due to the noise in K.
- So, by changing the value of S to (1, 1, 0). Hence SVD of E is given as,

$$E = USV^T$$

### 2.5 Extracting Camera Poses

- The camera pose depends on six parameters Rotation (roll, pitch, yaw) and Translation (x, y, z) of camera coordinate system with respect to the world coordinate system.

- Using Singular Value Decomposition of the E matrix to evaluate the Camera Positions,

$$E = UDV^T$$

- The camera pose can be estimated using equation

$$P = K\,R[I_{3\,x\,3} - C]$$

- The following correspondence of rotation and camera centers are extracted from E as shown below,

$$C1 \;=\; U(:,3) and\ R1 \;=\; UWV^T$$
$$C2 \;=\; -U(:,3)\ and\ R2 \;=\; UWV^T$$
$$C3 \;=\; U(:,3)\ and\ R3 \;=\; UW^T\,V^T$$
$$C4 \;=\; -U(:,3)\ and\ R4 \;=\; UW^TV^T$$

- One can obtain the corresponding four poses of R and C.
- The determinants of R and C were tested to see if they were positive to reduce camera pose error. In addition, from the four available poses, the right pose must be determined.

## 3. Rectification

- Stereo Rectification is reprojecting image planes onto a common plane parallel to the line between camera centers.
- A homography (3x3 transform) is applied to both input images to make horizontal epipolar lines.

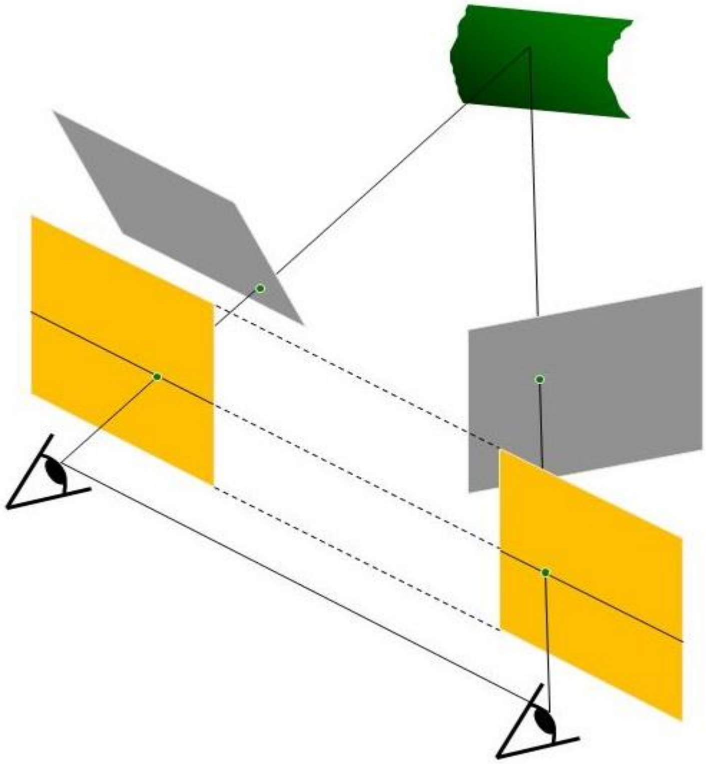


Figure- 5: Stereo Rectification

- It is convenient to if image scale lines are epipolar lines.
- It is achieved in OpenCV using the inbuilt function cv2.stereoRectifyUncalibrated which Computes rectification transforms for uncalibrated stereo camera.
- The feature calculates rectification transformations without understanding the cameras' intrinsic parameters or relative location in space.
- The output of image after stereo rectification is shown below,

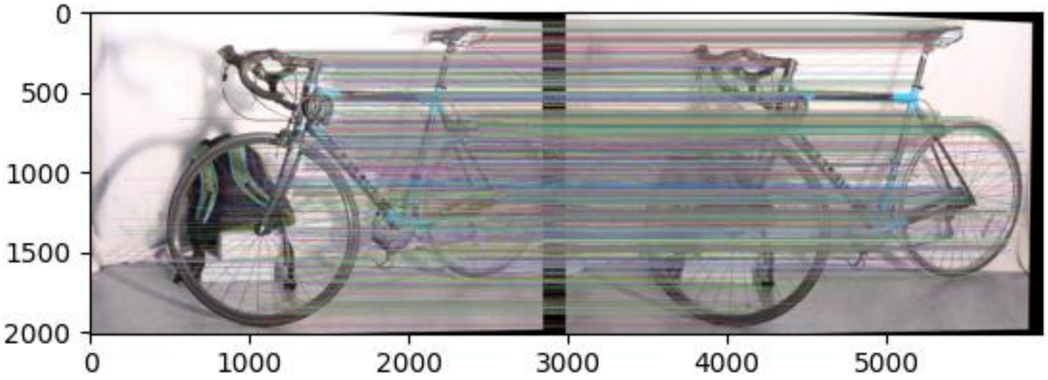


Figure-2: Epipolar lines with feature points for dataset-1

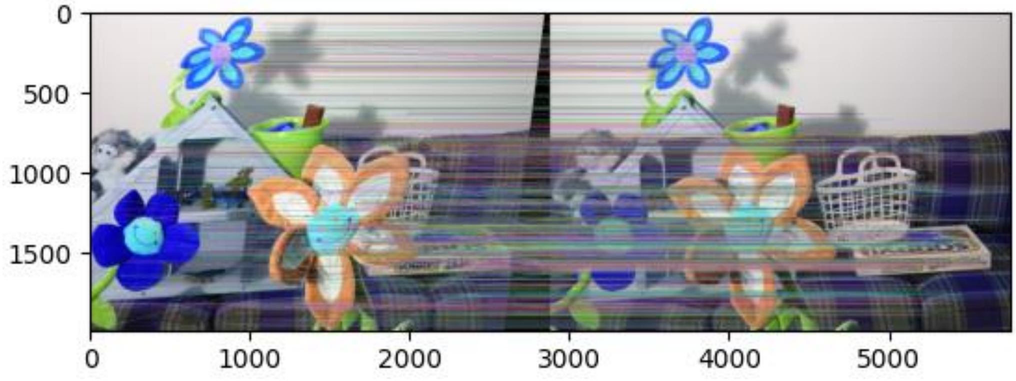


Figure-3: Epipolar lines with feature points for dataset-2

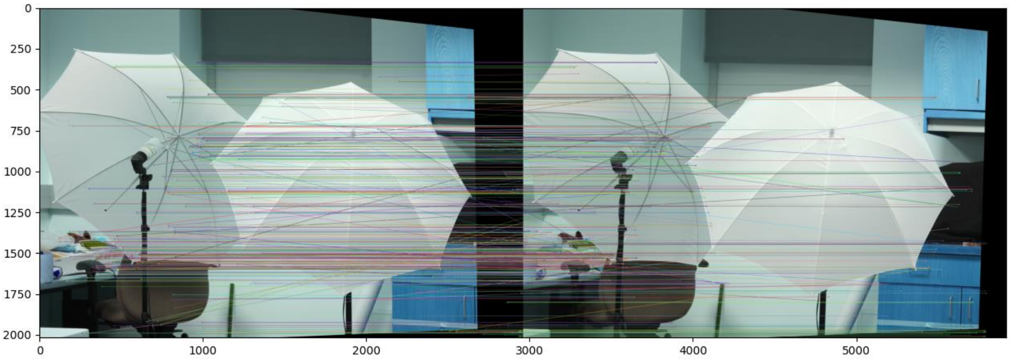


Figure-4: Epipolar lines with feature points for dataset-3

# 4. Correspondence

Owing to the cameras' differing viewpoints on the scene, objects in the field of view of stereo cameras can appear in different horizontal positions within the two photographs. The correspondence problem is defined as finding a match across these two images to determine. After getting rectified image, all scanlines are epipolar lines which converges at infinity.

## 4.1 Disparity

The amount of horizontal distance between the object in Image-1 and image-2 (*the disparity* d) is inversely proportional to the distance z from the observer. Disparities between the images are in the x-direction only (i.e., no y disparity).

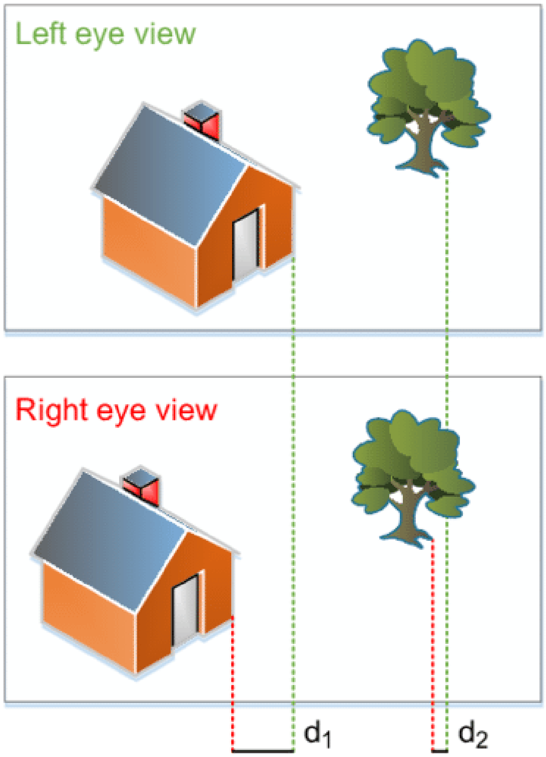


Figure-6: Example of disparity in two images.

### 4.2 Matching Window Concept

A correspondence will lie upon the epipolar line. Unfortunately, we cannot tell the precise pixel position of a match only because we know the Fundamental matrix between two images without looking for it along a 1d path.

For plotting the disparity map, I have used the Sum of absolute distance technique (SAD) followed by basic match algorithm.

The overview of pipeline for creating the disparity map is shown in figure below.

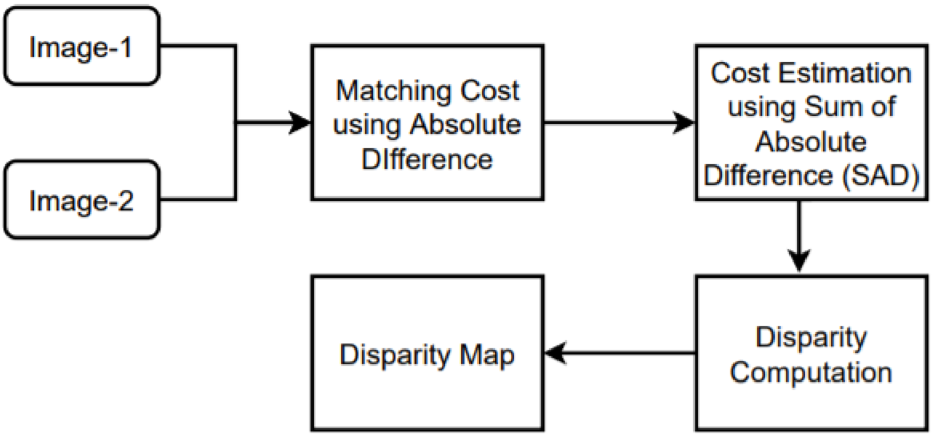


Image-1
Image-2
Matching Cost using Absolute DIfference
Cost Estimation using Sum of Absolute Difference (SAD)
Disparity Computation
Disparity Map

Figure-7: Overview of process for creating disparity map

Sum of absolute difference is Selecting the windows with the necessary dimension in the cost matrix and applying the variance between all components over the whole window results in the correspondence between two images. It is represented in equation as follows,

$$SAD(A,B) = \sum_{i,j} |A_{ij} - B_{ij}|$$

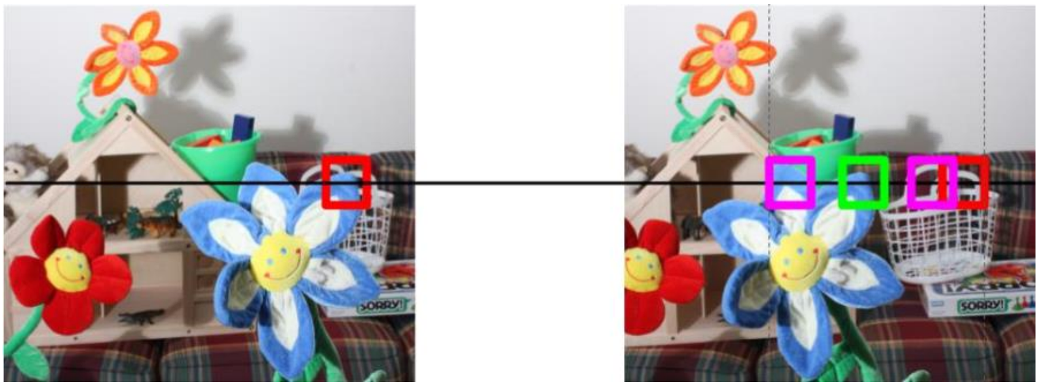


Figure-8: Matching window concept to find correspondence

For a given two images, the correspondence is found as follows,

- Selecting a small square block in the image 1(red block). Then search for the closest matching region in image 2.

- When search in image 2, start at the same coordinates as original.
- Computing the cost by comparing each block from image 1 and each block selected from search block in the image 2.
- Sliding the block on the image 2 by every pixel and aggregate the cost.
- Finding the closest matching from minimum cost which is shown by green box in image 2. Disparity is simply the distance between center of red block and green block.
- Repeat the matching process for each pixel in image 1 and calculating all the disparity values for image 1 index.
- Scaling those values in range of 0-255.
- Plotting those disparity values to create heat map, which is shown below,

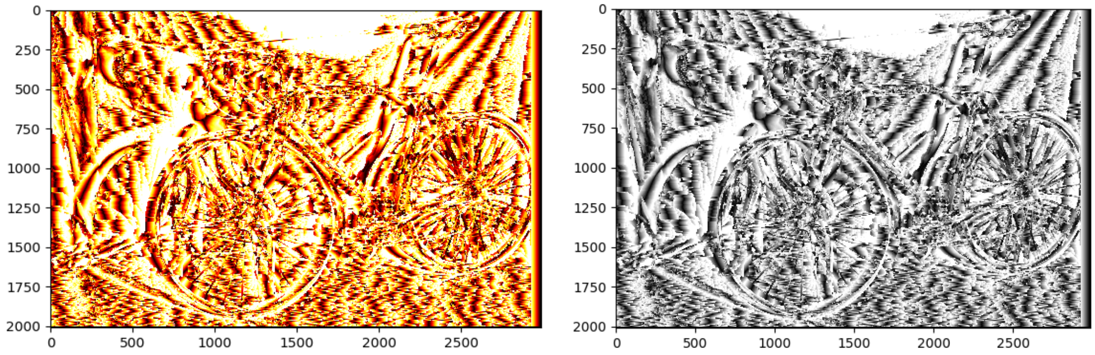


Figure-9: Disparity heat maps for Dataset-1

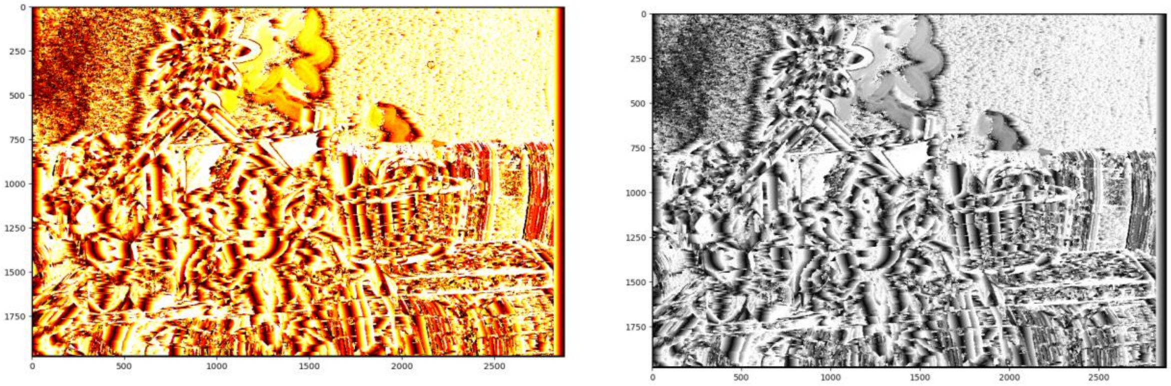


Figure-10: Disparity heat maps for Dataset-2

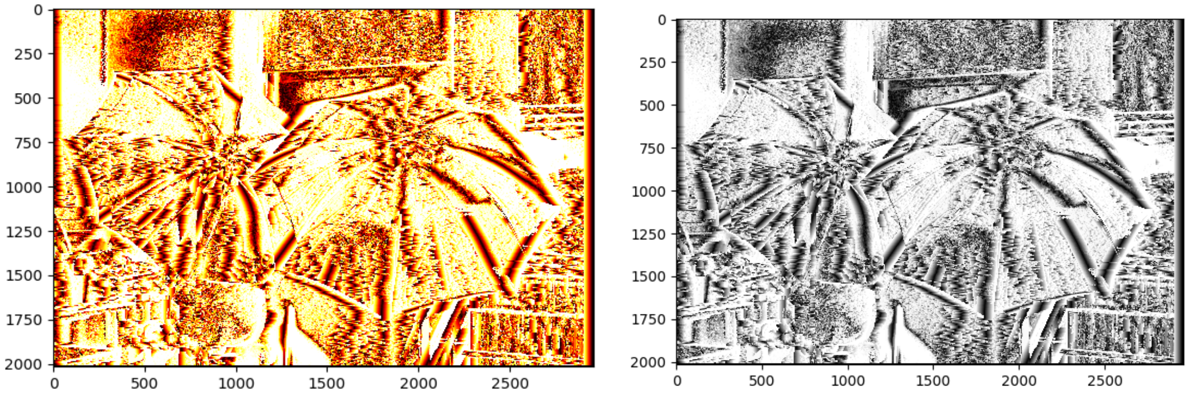
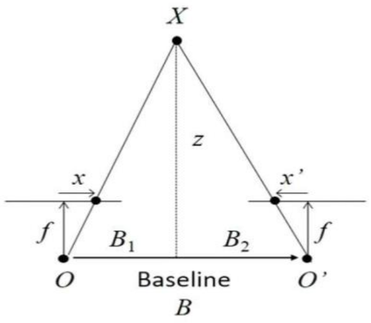


Figure-11: Disparity heat maps for dataset-3

## 5. Computing Depth Image

For every pixel depth can be computed using disparity. Considering a stereo vision system,



Depth can be calculated from the equation of disparity,

$$disparity = x - x` = (Baseline\ B) * \frac{(focal\ length\ \ f)}{depth\ (z)}$$

Depth map is generated by computing each pixel from the above equation. Depth maps play a significant role in estimation of 3D point from 2D points as well as various other computer vision application.

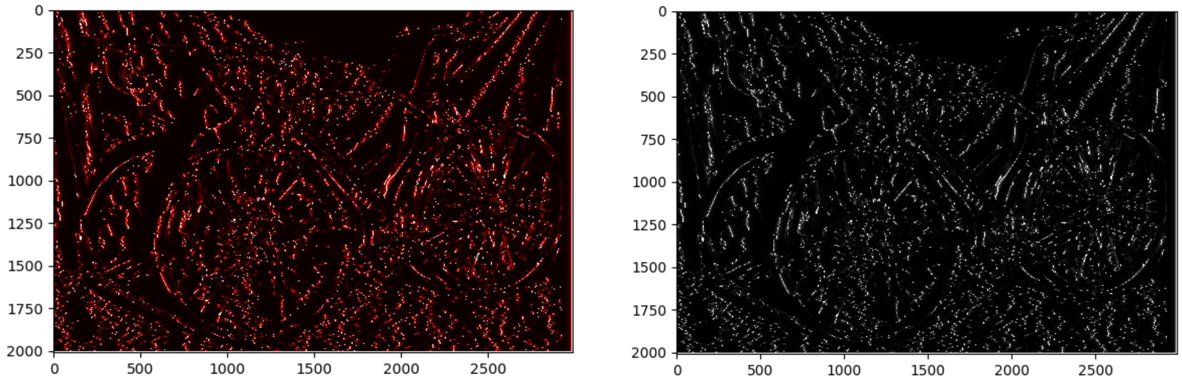


Figure-12: Depth Image for Dataset-1

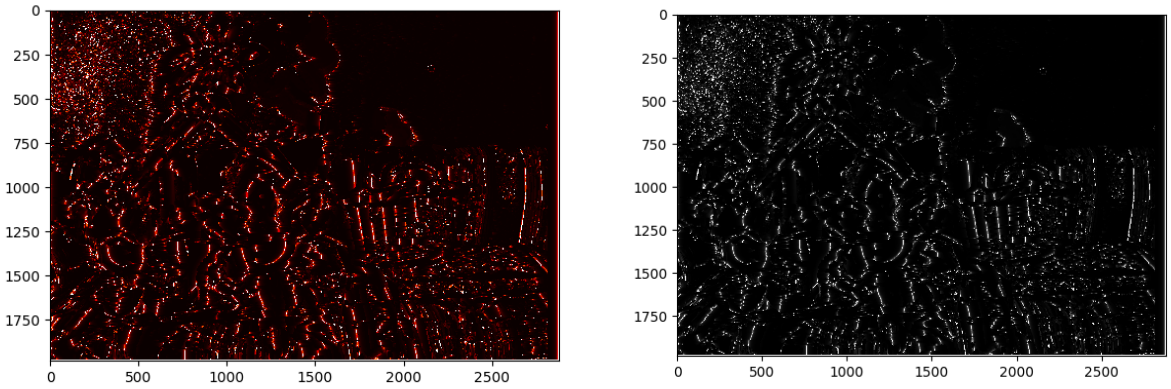


Figure-13: Depth map for dataset-2

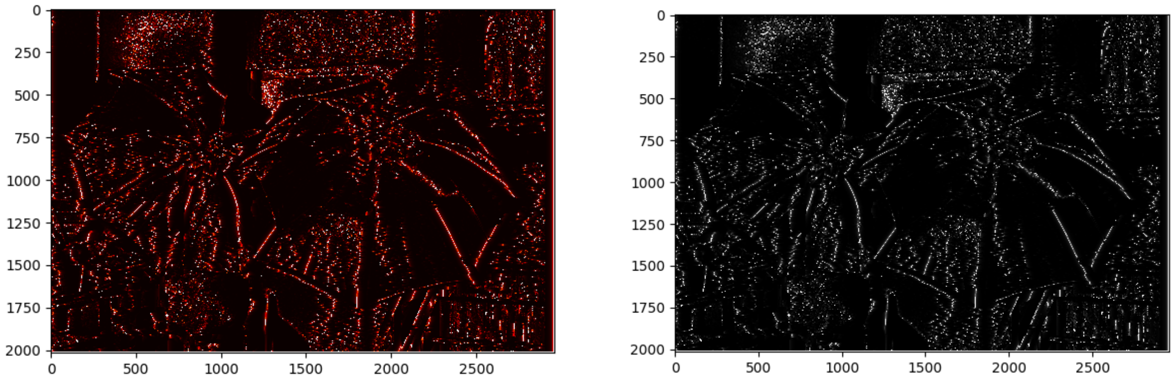


Figure-14: Depth map for dataset-3

## 6. Challenges or Problems faced

- The output can be made more robust by increasing the number of iterations used for RANSAC.

- For taking a small window size in disparity calculations, unique features of image are not distinguished correctly, but a large window size would result into less useful matches.
- Significant time was spent in understanding various window matching concepts such as SSD, SAD and NCC.

## 7. References

- “Feature Matching — OpenCV-Python Tutorials 1 documentation,” Readthedocs.io, 2013. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_matcher/py_matcher.html.
- Maged Aboali, Nurulfajar Abd Manap, Abd Majid Darsono, and Zulkalnain Mohd Yusof, “Performance Analysis between Basic Block Matching and Dynamic Programming of Stereo Matching Algorithm,” ResearchGate, Sep. 2017. https://www.researchgate.net/publication/320628497_Performance_Analysis_between_Basic_Block_Matching_and_Dynamic_Programming_of_Stereo_Matching_Algorithm/link/59f2e536a6fdcc1dc7bb3106/download
- “Buildings built in minutes - An SfM Approach,” *Github.io*, 2019. https://cmsc733.github.io/2019/proj/p3/#featmatch.

## 8. Google Drive Link

- https://drive.google.com/drive/folders/1DhUD1Hi0LJcKv7mtX8iXTvTnw12ok5dI?usp=sharing