# CSE 598 Perception in Robotics Project 2a Report: Camera Model and Stereo Depth Sensing

Siddharth Das <sdas100@asu.edu>

**Project Execution details:**
**Working Directory: project2a/**
**Dependencies:**

| | |
|---|---|
| matplotlib | 3.3.4 |
| numpy | 1.20.1 |
| opencv-python | 4.2.0.34 |

**To run the project in Ubuntu terminal:**
**Step 1:** unzip Siddharth_Das_project_2a_resources.zip
Step 2: cd Siddharth_Das_project_2a_resources/project_2a
Step 3: pip install -r requirements.txt
Step 4: python code/task_[task_number]/task_[task_number].py
(Replace [task_number] with the actual task number)

**Task 1: Pinhole camera model and calibration**

First, the images were imported as np.arrray using opencv imread() function then the image points were extracted using findChessboardCorners() for both left and right view images(separately) in two lists. Then using calibrateCamera() the camera intrinsics and distortion coefficients were found out. Later these intrinsics and distortion coefficients left_2.png and right_2.png images were distorted.

Output:

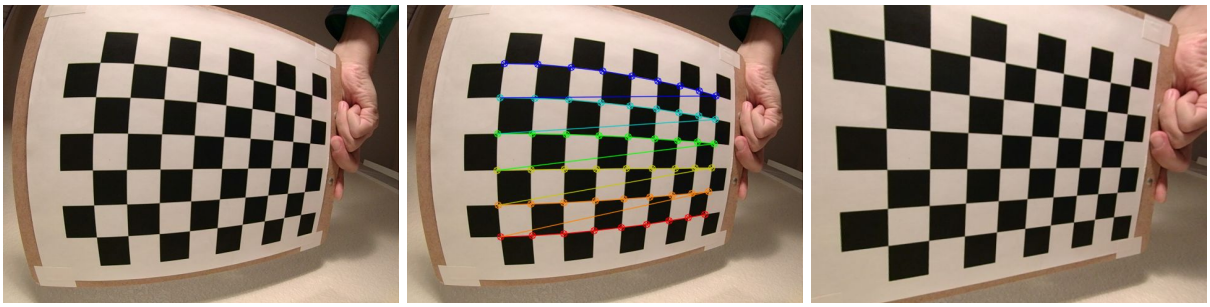

Figure 1(a): The original calibration board pattern (left), the same pattern with corner points annotation (middle), and the undistorted pattern (right) for left_2.png.

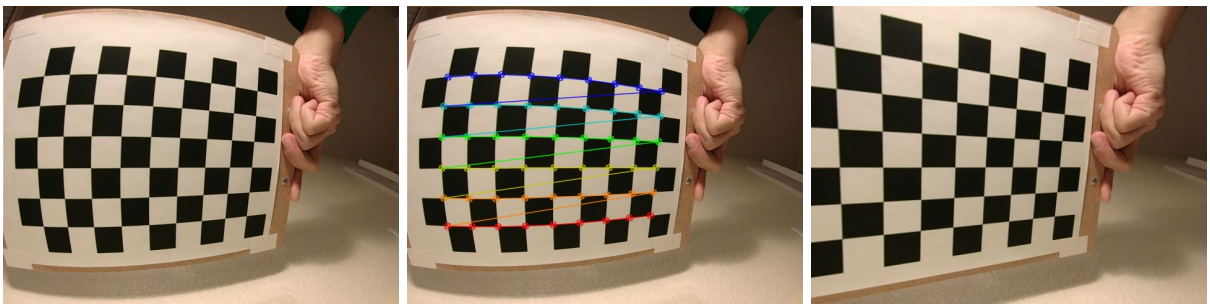

Figure 1(a): The original calibration board pattern (left), the same pattern with corner points annotation (middle), and the undistorted pattern (right) for right_2.png.

**Task 2: Stereo calibration and rectification**

Initial steps of Task 1 were repeated to get the image points. Then the extrinsic parameters were computed using stereoCaliberation() function passing the world points, image points and previously saved intrinsic values from task 1. Then stereo rectification was done and triangular points were computed.

**Output**:

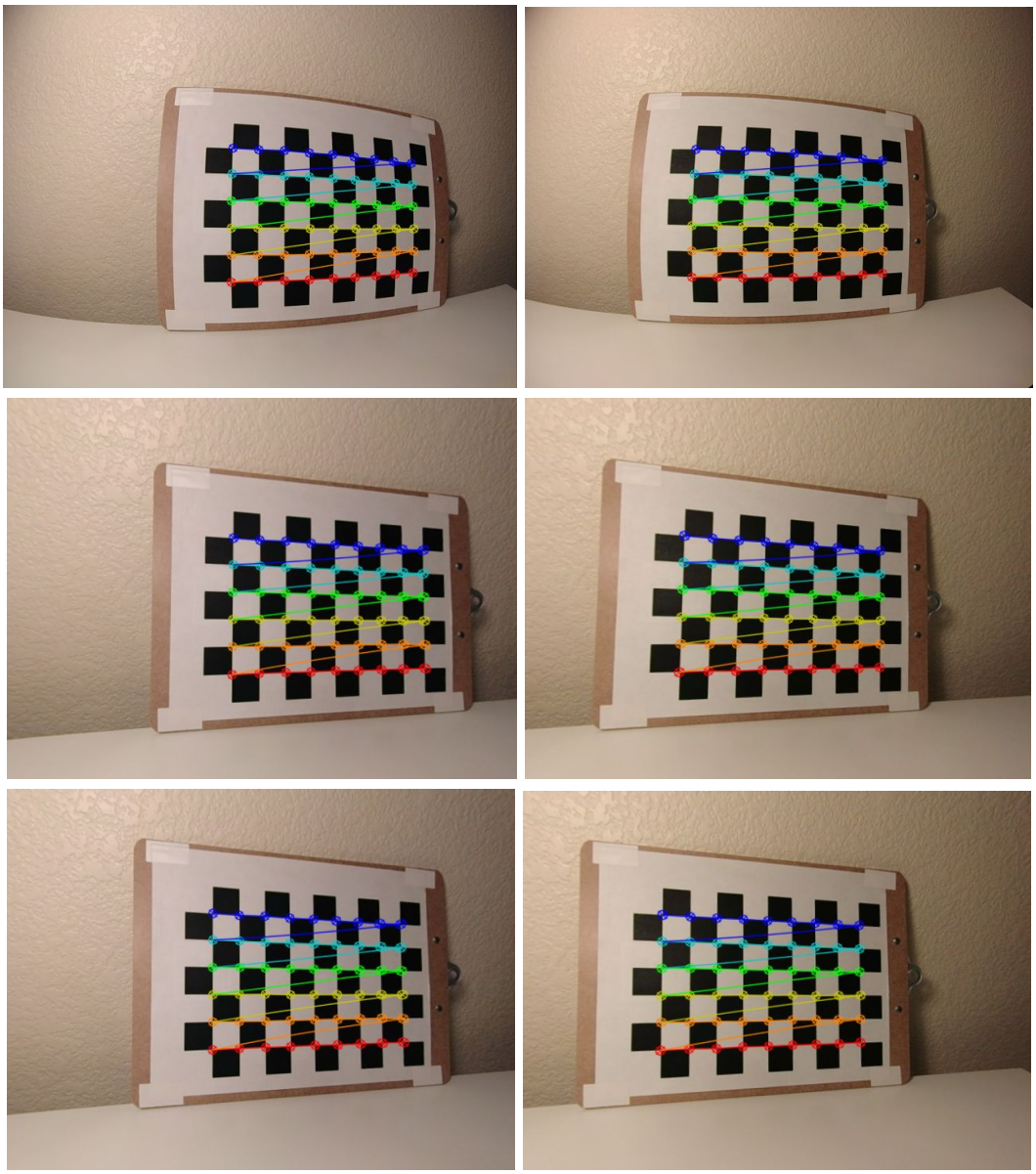

Figure 2: Left(left) view and right view(right), original (top), undistorted but not rectified (middle), undistorted and rectified (bottom).


Figure 3: Camera pose before rectification (left) and after rectification (right).

**Task 3: Sparse depth triangulation**

Output: First the images were undistorted and rectified using parameters saved from task 2, then the feature key points were detected and local maxima points were selected for matching. After matching only the points satisfying epipolar constraints were passed for triangulation and plotted before and after converting coordinates to actual scale.

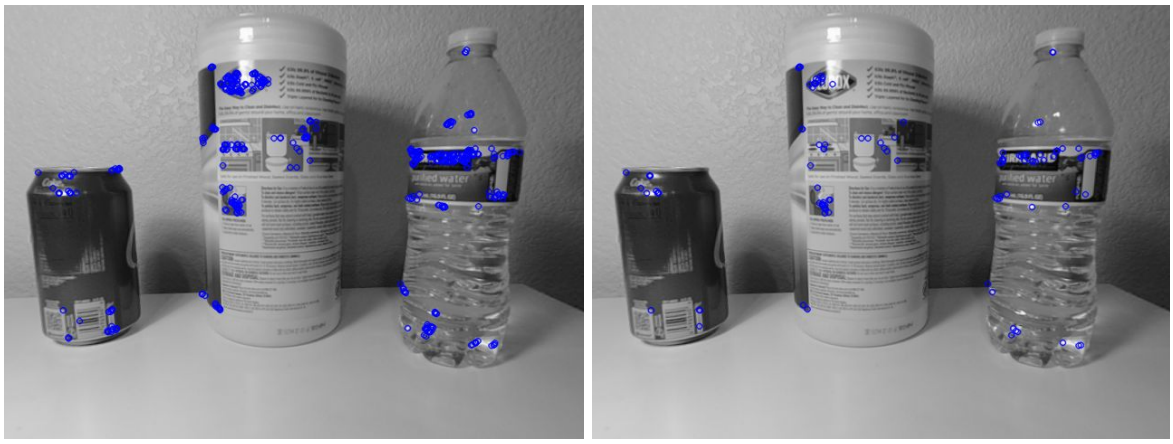

Figure 4: An example of all detected feature points (left) and selected local maxima feature points with a six-pixel radius (right).

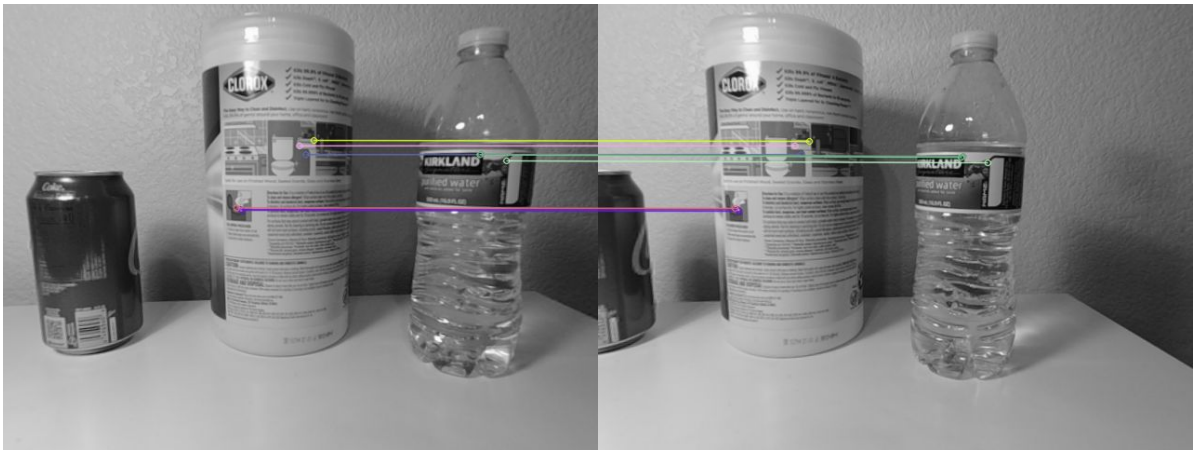


Figure 5: An example of selected matches of feature points on the two views.

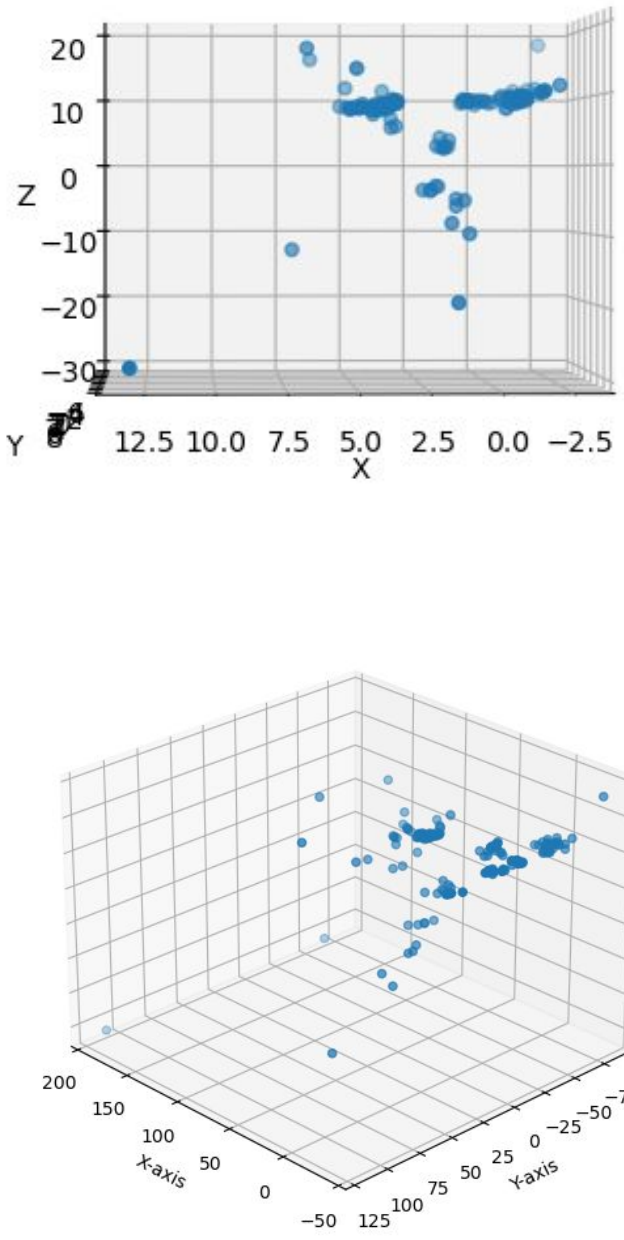


Figure 6: Sparse depth triangulation results: (top) obtained 3D points, (bottom) 3D points(in actual scale) on a cylinder surface from a certain view.

**Task 4: Dense depth triangulation.**

Dense depth triangulation results for 2 pairs of images-

- Pair 1: left_4.png and right_4.png
- Pair 2: left_9.png and right_9.png

Output:

Figure 7: Rectified images from the stereo camera- left camera view(left_4.png) (left), and right camera view(right_4.png) (right).

Figure 8: Disparity map for 4th pair of images(left_4.png and right_4.png)

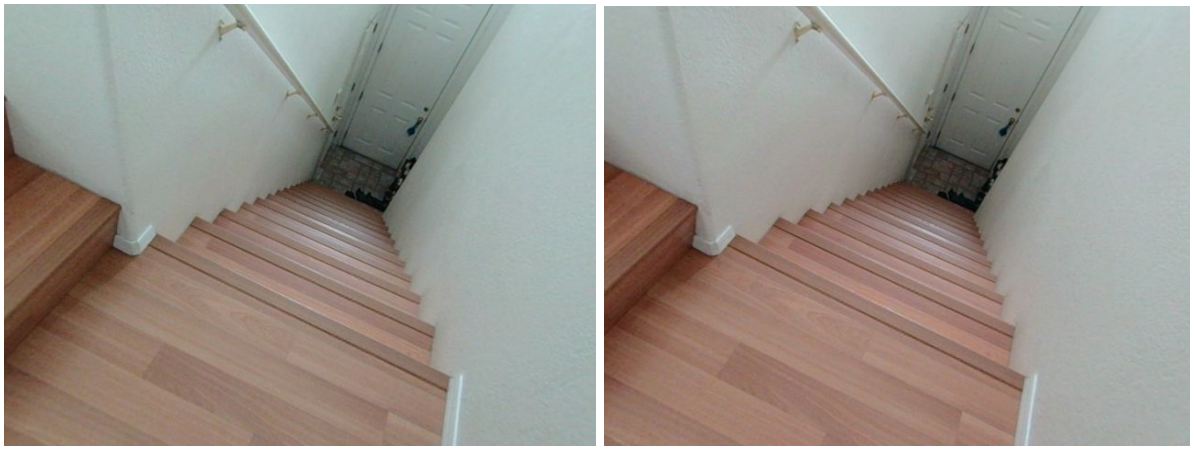

Figure 9: Rectified images from the stereo camera- left camera view(left_9.png) (left), and right camera view(right_9.png) (right).

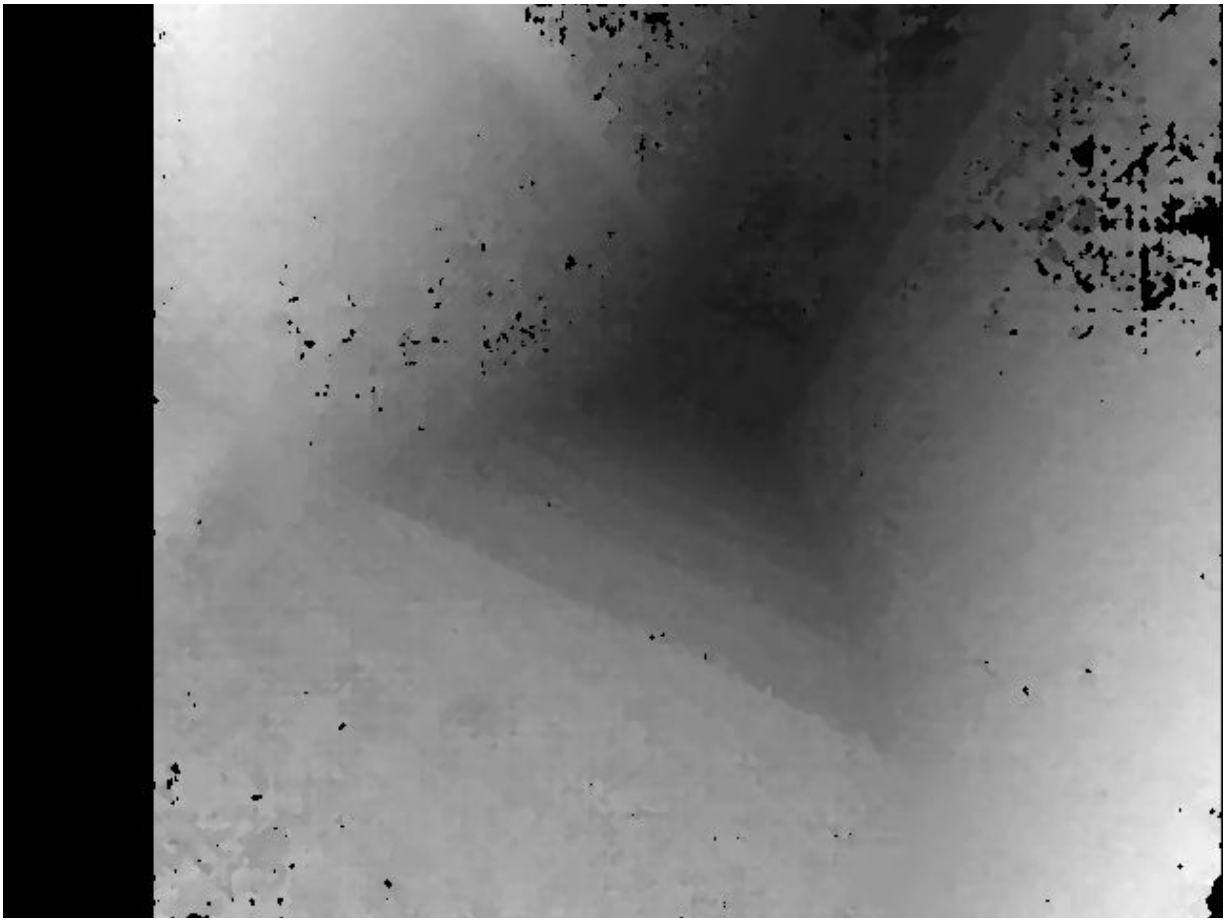

Figure 10: Disparity map for 4th pair of images(left_9.png and right_9.png)