3D Scene Reconstruction using Stereo Vision

Harshith Mohan Kumar*§, Dhruval P B*§, Girivinay Padegal*§, Akash Mehta*§
*Department of Computer Science and Engineering, PES University,
Bengaluru, India
{harshithmohankumar, dhruvalpb, pes1ug19cs315}@pesu.pes.edu

***Abstract*—In this paper, we look at the problem of reconstructing a scene in 3D using multiple stereo image pairs. We achieve this by predicting how the camera setup moved while generating the data with the help of a monocular SLAM algorithm called TwitchSLAM. We obtain point clouds of every stereo pair by back projecting the disparity map calculated by a stereo depth perception algorithm called Efficient Large Scale Stereo (ELAS). The predicted camera trajectory is then used it to translate and rotate these pointclouds which are then stacked together and rendered at once. The results of our experiments show that the proposed method can generate high-fidelity 3D renders of the scene.**

***Index Terms*—3D Scene Reconstruction, Stereo vision, Point clouds, Monocular SLAM**

I. Introduction

In recent years, there has been a strong shift towards using cheaper stereo vision cameras for computing depth over the more accurate but expensive sensor units. The cost of these depth based sensors such as the LiDAR, Radar, etc. far outweigh the accuracy which they bring to the table. Likewise, humans are able to perceive depth and the environment around us solely based on our visual system. The huge advancements in machine learning and deep learning has sparked a new revolution in an attempt to replicate the accuracy of LiDAR units by using only vision. One particular application of stereo based depth perception which is heavily under research is the acquisition of the 3D environment structure from a given visual input, such as a picture. This happens to be one of the most essential challenges for natural and artificial visual perception systems.

The core problem with this topic is that visual input is frequently the result of a projection mapping from a three-dimensional world to a two-dimensional manifold. This transformation does not have a one-to-one correspondence, i.e. it cannot be inverted. As a result, mapping back to the 3D environment, also known as the 3D reconstruction task, is an inverse issue. Nonetheless, humans and other living organisms can generate relatively accurate representations of the underlying 3D structure even when just given a single visual input, implying that they can recover the information lost during the projection process. The important resource for accomplishing this is prior experience, and this is our fundamental reason for using machine learning approaches to handle the 3D reconstruction issue for artificial systems such as robots using just single photos. Such a technology has several potential uses. While most existing techniques to creating 3D environment models rely on the integration of many photos taken from various perspectives.

§These authors contributed equally to this work

II. Previous Work

A. ORB SLAM2

ORB-SLAM2 [1] is a continuation of ORB-SLAM [2], where the main components for tracking, mapping and place recognition are reused. This model has the capability of producing accurate trajectory estimation with metric scale. It extends on the previous implementation by including stereo and RGB-D as alternative inputs.

The inclusion of stereo input had many benefits, particularly in improving the estimated trajectory. When reconstructing the trajectory of a moving camera, the scale of the environment, is obtained by accumulating relative scale estimates over sequences of frames. Errors in scale accumulate over time and this is commonly known as scale drift [3]. Monocular ORB-SLAM suffers from severe scale drift which has been notable especially at turns. The proposed stereo vision on the other hand is able to estimate the true scale of the trajectory and map it accordingly without scale drift [1]. The difference can be visualized in Fig. 1.

ORB-SLAM2 has been evaluated on three popular datasets and one of which is the KITTI dataset [4]. This dataset which contains sequences of stereo images is used to create points using the keyframe. Unlike a monocular system, the model only requires one stereo keyframe. Although ORB-SLAM2 is a wonderful model for stereo images, it lacks support for other camera types such as non-overlapping multi-camera, fisheye or omni-directional cameras support [1]. These are areas in which the authors plan to expand to for their future work. In our work we wish to utilize the trajectory mapping capabilities of ORB-SLAM2 and merge it with ELAS to produce a three dimensional representation of the environment. This is discussed in greater detail within Section IV.

B. Multiple Spherical View Reconstruction

Stereoscopic image rectification is a widely studied topic that provides the ability to estimate 3D depth from 2D input images. It is a field that has received a lot of notice but using multiple views (more than 2) is something that has lacked thorough exploration. The hypothesis that the usage of multiple spherical views actually assists in more accurate depth estimation and disparity mapping is explored in [5].

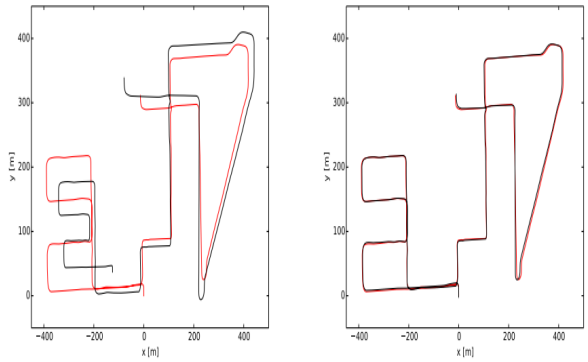


Fig. 1: Estimated trajectory (black) and ground-truth (red) in KITTI 08. Left: monocular ORB-SLAM [2], right: ORB-SLAM2 (stereo). Monocular ORB-SLAM suffers from severe scale drift in this sequence, especially at the turns. In contrast the proposed stereo version is able to estimate the true scale of the trajectory and map without scale drift [1]

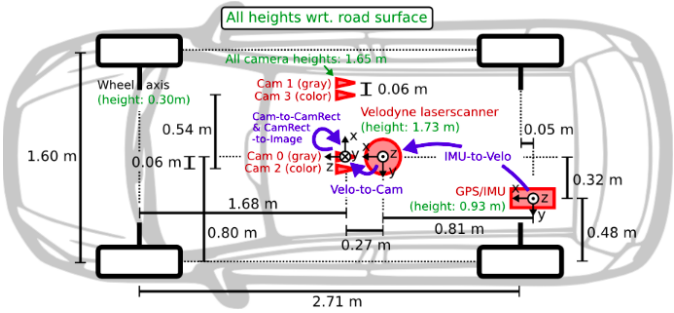


Fig. 2: Sensor setup used for generating the KITTI dataset

A 5-viewpoint camera capture system is utilised with each view point having a pair of vertically stacked spherical cameras. Optical flow estimation is a common structure-from-motion principle that allows us to estimate the x and y components of motion of objects in a two dimensional scene. First, maximally stable extreme regions (MSERS) are averaged in the reference image to accentuate depth, then SIFT keypoints are matched with the reference image and box filtering is performed at locations of SIFT keypoints.

A drawback of this approach is that spherical camera images are high resolution and need to be downsampled which lead to occurrence of artifacts. An additional observation is that the usage of multiple cameras viewpoints has little impact on the accuracy of 3D reconstruction while drastically reducing algorithm speed; an observation which is taken into consideration. Also, it is hypothesised that the usage of a point cloud reconstruction would result in much better results than overlaying the image onto the inverse of the depth map on 3D axes; another observation that is utilised in our approach.

III. Dataset

We have used the KITTI dataset that has been made by equipping a standard station wagon with two high-resolution color and grayscale video cameras. In total, they recorded 6 hours of traffic scenarios at 10-100 Hz using a variety of sensor modalities such as high resolution color and gray scale stereo cameras, a Velodyne 3D laser scanner and a high-precision GPS/IMU inertial navigation system. The scenarios are diverse, capturing real-world traffic situations and range from freeways over rural areas to inner city scenes with many static and dynamic objects. Accurate ground truth is provided by a Velodyne laser scanner and a GPS localization system. The data is calibrated, synchronized and timestamped, and both provide rectified and raw image sequences are provided. We chose to experiment with the raw image sequences. Fig. 2 shows the placement of the sensors used for generating this dataset.

IV. Proposed Solution

The design of our implementation is shown in Fig. 3. The system receives a images from a pair of calibrated cameras as input. We perform Monocular SLAM using twitchslam on the left image and the predicted camera orientations are stored. A Disparity map is generated using stereo depth perception algorithm named ELAS. 3D point clouds are generated by back projecting the disparity maps and are saved. The saved camera orientations are then used to translate and rotate the saved 3D pointclouds. These updated pointclouds are then stacked together and rendered at the same time to obtain a 3D reconstruction of an entire scene.

A. Point Cloud Generation

1) Disparity Generation using ELAS: Efficient LArge-scale Stereo [6] abbrievated as ELAS is a Bayesian approach to stereo matching that is able to compute accurate disparity maps of high resolution images at frame rates close to real time without the need for global optimization. It overcomes the problems faced by algorithms that employ local correspondence such as border bleeding effects and even outperforms by over three orders of magnitude the global correspondence algorithms that have high computational and memory requirements in the face of high resolution images.

This approach makes use of the fact that in spite of many correspondences being highly ambiguous, some of them can be robustly matched. These reliable 'support points' contain valuable prior information that can be used to estimate the remaining ambiguous disparities. The image coordinates of the support points are then used to create a 2D mesh via Delaunay triangulation. A prior is computed to disambiguate the matching problem, making the process efficient by restricting the search to plausible regions. In particular, this prior is formed by computing a piecewise linear function induced by the support point disparities and the triangulated mesh.

Support points are matched on a regular grid in an efficient and effective manner by using the $\ell 1$ distance between vectors formed by concatenating the horizontal and vertical

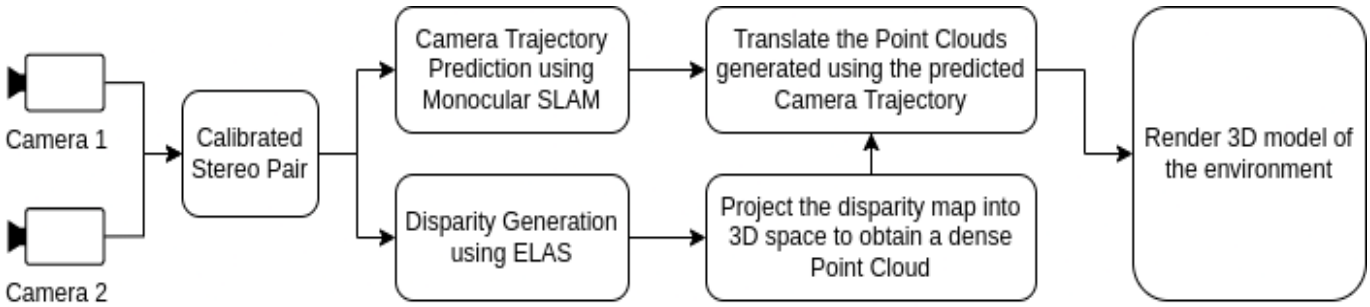


Fig. 3: Pipeline

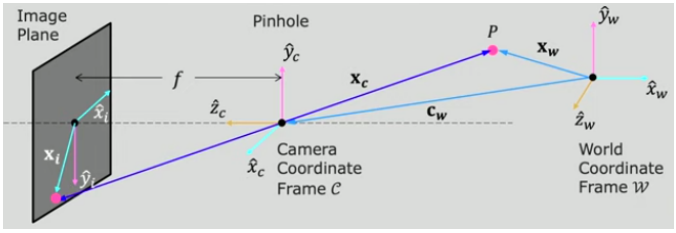


Fig. 4: Forward Imaging Model: 3D to 2D

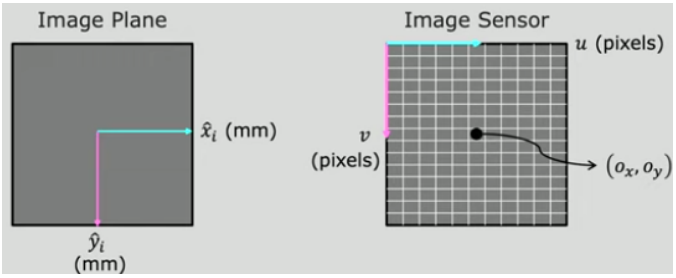


Fig. 5: Image Plane to Image Sensor Mapping

Sobel filter responses of 9 × 9 pixel windows. To ensure robustness, consistency is imposed, i.e, correspondences are retained only if they can be matched from left-to-right and right-to-left. A probabilistic generative model is designed with an image likelihood and a prior defined to be proportional to a combination of a uniform distribution and a sampled Gaussian. Finally the disparities are computed by using maximum a-posteriori (MAP) estimation while taking into consideration all observations on the right image that occur on the epipolar line of the corresponding observation on the left image. We have utilised an implementation of the ELAS algorithm that has been accelerated using CUDA [7] and OpenMP [8].

2) Back Projection: Once the disprity map is obtained, we need to back project this into 3D space using the camera calibration parameters to obtain a point cloud of the scene. Calibration parameters of a camera consist of Extrinsic parameters (which denote the position and orientation of the cameras with respect to the world coordinate frame) and Intrinsic parameters (which refer to how the camera maps the points in the world onto its image plane via perspective projection). From the Forward Imaging Model as shown in Fig. 4, any point in the world, denoted by (x_w, y_w, z_w) undergoes two transformations before it maps to a pixel on the image generated by the camera. The first is 3D to 3D transformation, a rotation and translation from the world coordinate frame to the camera coordinate frame, (x_c, y_c, z_c). The next transformation is called perspective projection and is shown in the Eqn. 1

$$(x_i, y_i) = (f \times \frac{x_c}{z_c}, f \times \frac{y_c}{z_c}) \tag{1}$$

Here, (x_i, y_i) is the coordinates of the projection of the real world point onto the image plane in terms of millimeters (mm) (same units as that of the camera coordinate frame). However, in case of digital cameras, the image plane is a digital image sensor consisting of pixels. Hence, the pixel coordinates (u, v) can be obtained as shown in Eqn. 3

$$(u, v) = (f_x \times \frac{x_c}{z_c} + o_x, f_y \times \frac{y_c}{z_c} + o_y) \tag{2}$$

Here, f_x is the effective focal length of the camera in the x direction and f_y is the effective focal length of the camera in the y direction. $[o_x o_y]$ is the Principle point, the point where the optical axis pierces the sensor. The reason behind having two effective focal lengths is that the pixels of an image sensor may not be square, leading to uneven pixel densities along the x and y directions. Fig. 5 shows the mapping from the Image Plane (mm) to the Image Sensor (Pixels).

f_x, f_y, $_{ox}$ and o_y are the camera's Intrinsic parameters that represent its internal geometry. Since these equations are non linear, we make use of Homogeneous Coordinates by introducing a fictitious third coordinate $\widetilde{w} \neq 0$ such that,

$$(u, v) = (\frac{\widetilde{u}}{\widetilde{w}}, \frac{\widetilde{v}}{\widetilde{w}}) \tag{3}$$

Hence, we can represent a 2D point (u, v) in Homogeneous coordinates as $(\widetilde{u}, \widetilde{v}, \widetilde{w})$. Similarly, a 3D point (x, y, z) can be represented in Homogeneous coordinates as $(\widetilde{x}, \widetilde{y}, \widetilde{z}, \widetilde{w})$. Now, rewriting Eqn. 1 in Homogeneous Coordinates, we obtain,

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \widetilde{u} \\ \widetilde{v} \\ \widetilde{w} \end{bmatrix} = \begin{bmatrix} z_c \times u \\ z_c \times v \\ z_c \end{bmatrix} = \begin{bmatrix} f_x \times x_c + z_c \times o_x \\ f_y \times y_c + z_c \times o_y \\ z_c \end{bmatrix} \tag{4}$$

The RHS can be represented as a matrix multiplication of the Intrinsic matrix (containing the intrinsic parameters of the camera) and the Homogeneous coordinates of the point in the camera coordinate frame as shown in below.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \tag{5}$$

We can represent the transformation from the world coordinate frame to the camera coordinate frame in Homogeneous coordinates as follows,

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} r_1 1 & r_1 2 & r_1 3 & t_x \\ r_2 1 & r_2 2 & r_2 3 & t_y \\ r_3 1 & r_3 2 & r_3 3 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{6}$$

Here, the 4x4 matrix in the RHS is termed the Extrinsic matrix which consists of a 3x3 rotation matrix and a 3x1 translation matrix. Now, in order to find the real world coordinates of a point corresponding to a pixel in an image, we need to perform the inverse of the aforementioned transforms. However, projection from 2D to 3D gives us a ray on which the point can lie in 3D space. We need atleast another ray that intersects this to find the actual location of the point in 3D. Hence, we add another camera as shown in (7), where b is the baseline length between the two cameras, (u_l, v_l) is the point in the left camera and (u_r, v_r) is the point in the right camera.

$$\begin{aligned} (u_l, v_l) &= (f_x \times \frac{x}{z} + o_x, f_y \times \frac{y}{z} + o_y) \\ (u_r, v_r) &= (f_x \times \frac{x-b}{z} + o_x, f_y \times \frac{y}{z} + o_y) \end{aligned} \tag{7}$$

When we solve these equations for the point of intersection, we obtain,

$$(x_c, y_c, z_c) = (\frac{b(u_l - o_x)}{(u_l - u_r)}, \frac{bf_x(v_l - o_y)}{f_y(u_l - u_r)}, \frac{bf_x}{(u_l - u_r)}) \tag{8}$$

Here, the denominator $(u_l - u_r)$ is known as the disparity $(d(x, y))$ and the 3D depth (z) of a point is inversely proportional to its disparity. We can then transform the coordinates of the point in the camera-system's coordinate plane, (x_c, y_c, z_c) to that of the world coordinate frame by multiplying it with the inverse of the Extrinsic Matrix, i.e, the inverse of Eqn. 6.

B. TwitchSlam

TwitchSLAM is a by-product of a fun twitch stream hosted by George Hotz. During this stream, he implemented moncoular simultaneous localization and mapping (SLAM) from scratch using python. The primary sensory input to the system in monocular SLAM is a single camera that moves freely through its environment. George Holtz made use of the KITTI dataset to test the output of his implementation. However as discussed before, there is a severe limitation to obtaining depth from monocular input as compared to stereo. On the other hand, the optical flow algorithm implemented performs fairly well. Therefore in our implementation we utilize the Twitch-SLAM implementation to extract the position and rotational orientation of the camera within a given sequence of frames. This information is stored as a 4x4 matrix by specifying rotations and translations in 3 dimensions using homogeneous coordinates.

Rotation is specified as a 3 dimensional matrix. Rotation can be along any of the three axes (X,Y, or Z) and therefore each can be specified as a canonical rotation matrix. The following matrices demonstrate the representation for each:

$$ROT(Z, \theta) = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{9}$$

$$ROT(X, \theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{bmatrix} \tag{10}$$

$$ROT(Y, \theta) = \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix} \tag{11}$$

Finally, the 4x4 matrix corresponding to each frame is stored as a numpy array and is extracted in another script and plotted along with the disparity data obtained from the ELAS implementation.

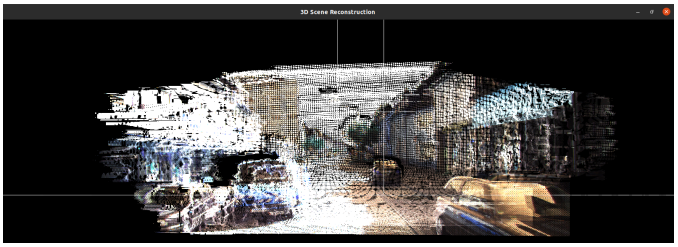
C. Rendering the scene in 3D

We have used PyQt for rendering the 3D scene. The scene is reconstructed from multiple consequent pointclouds that have been translated and rotated based on the camera orientation predicted by twitchslam. Fig. 6 shows multiple views of the 3D reconstruction of a scene from the Kitti Raw dataset.

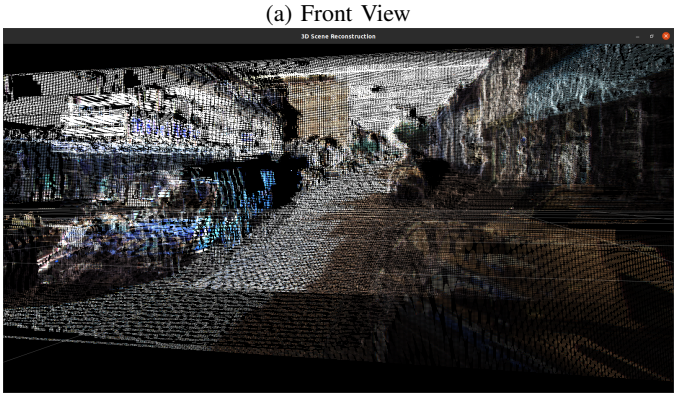
V. Evaluation Methodology

In recent years, full-reference (FR) point cloud evaluation process (PCQA) has made significant progress. However, because acquiring the reference point cloud in many circumstances is challenging, no-reference (NR) approaches seem to be a research hot spot. Due to the unavailability of a large-scale subjective point cloud collection, few studies on NR objective quality indicators are done. Furthermore, due to the uniqueness of the point cloud format, it is unable to simply employ blind image quality evaluation (IQA) methodologies to forecast the quality scores of point clouds. in this case the proposed method would be to calculate the density and compare the sparseness of the points layered into each point cloud with respect to each algorithm.

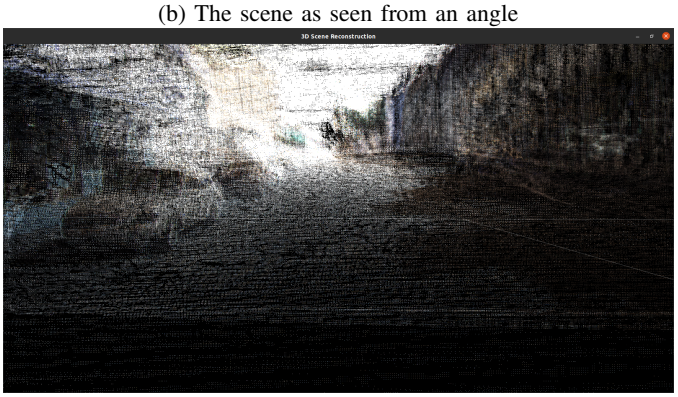
It is known that the ELAS algorithm generates a very detailed disparity map which when back projected, presents a point cloud of much higher density and resolution. Therefore we can calculate the increase in density of the point cloud using a density matrix and give a definite estimate of how much the quality of the point cloud has increased using the



(a) Front View



(b) The scene as seen from an angle



(c) Zoomed view (seeing cars 10's of metres away)

Fig. 6: Multiple views of the 3D reconstruction of a scene from the Kitti Raw dataset

ELAS back projection over the Twitchslam output. The density can either be the density is estimated by counting for each point the number of neighbors N (inside a sphere of radius R) or simply estimated by determining the distance to the nearest neighbor, which is generally much faster. This distance is considered as being equivalent to the above spherical neighborhood radius R (and N = 1). In addition to this we can also evaluate the uniformity of these densities to get a better grasp of the improvement.

The concept of a local directional density measure that may be derived organically inside the point cloud, that is, without any further knowledge of the geometry notwithstanding the supplied point samples. There are three approaches to this, The first is a continuation of the work presented in [C. Lange and K. Polthier. Anisotropic smoothing of point sets.] and builds on the covariance matrix. The second works with a unit circle on the tangent plane and the corresponding segments

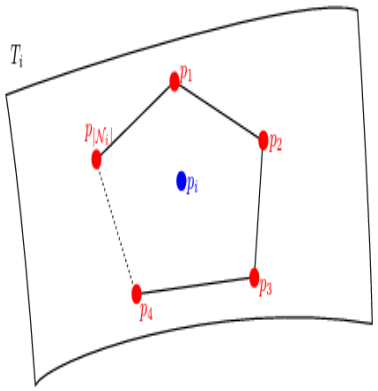


Fig. 7: Neighborhood of a point P_i on T_i given as a regular N_i-gon.

induced by projections of neighboring points. Finally, the third measure utilizes smooth basis functions. We will be using the first method, covariance matrices due to their simplicity., the authors present an approach to estimate the density of a point cloud P in a given direction from a point pi P. They propose to use the eigenvalues and the eigenvectors of the covariance matrix built on a neighborhood Ni of pi.

VI. Conclusions

The method proposed in this report is a unique approach to solving 3D environment reconstruction. Although direct approaches exist which utilize the power of various deep learning models, we have decided to combine the power of two separate models. We have divided the task into two distinct sections which are trajectory mapping and disparity map generation. Although Monocular SLAM approaches like ORB-SLAM2 and twitchslam produce 3D reconstruction, they are very sparse as they take into consideration only the Key-Points in subsequent frames. When we use a dense correspondence approach like ELAS for disparity calculation and later back project the same onto 3D space, we obtain a much denser point cloud, thus boosting the fidelity of the scene reconstruction. Although we have provided two very specific algorithms for our approach, we want to keep our options open for iteration while conducting our experiments.

VII. Work Split

We tried our best to obtain our own ground truth data by using the ZED Camera sensors from the Aeolus lab. However due to permission issues with the mechanical department we were denied access to the sensors midway through our testing. After which we spent a lot of time to integrate the outputs of ELAS and TwitchSLAM to generate the 3D renders. Vinay worked on camera calibration and feeding the input to the pipeline. Akash initially worked on the ZED cameras by setting up and using the provided SDK to obtain depth estimation from the ZED Mini camera. Harshith worked on

manipulating the TwitchSLAM library to obtain the angular orientation and position of the camera. Dhruval contributed by developing the code to extract the disparity map from using the ELAS algorithm, generate point clouds by back projecting the same and rendering the merged pointclouds. Overall the entire team collaborated thoroughly and spent a great deal of time and effort into this project.

REFERENCES

[1] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, oct 2017. [Online]. Available: https://doi.org/10.1109%2Ftro.2017.2705103

[2] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, oct 2015. [Online]. Available: https://doi.org/10.1109%2Ftro.2015.2463671

[3] T. Botterill, S. Mills, and R. Green, "Correcting scale drift by object recognition in single-camera slam," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1767–1780, 2013.

[4] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[5] I. Igbinedion and H. Han, "3d stereo reconstruction using multiple spherical views," 2019.

[6] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," in *Asian conference on computer vision*. Springer, 2010, pp. 25–38.

[7] NVIDIA, P. Vingelmann, and F. H. Fitzek, "Cuda, release: 10.2.89," 2020. [Online]. Available: https://developer.nvidia.com/cuda-toolkit

[8] OpenMP Architecture Review Board, "OpenMP application program interface version 3.0," November 2015. [Online]. Available: https://www.openmp.org/wp-content/uploads/openmp-4.5.pdf