Final Project Report: Computer Vision

3D Motion and Structure Reconstruction from Dense Optical Flow

Sudipta Paul[1] and Nowaz Rabbani[2]

[1]RIN: 662026666
[2]RIN: 662011081

1 Introduction

3D motion estimation and structure reconstruction of moving objects is one of the challenging tasks in computer vision [1, 2]. Methods implemented in literature for the task can be divided into two categories: Sparse motion estimation and structure reconstruction (e.g. Kalman filtering and factorization), Dense motion estimation and structure reconstruction (e.g. from dense optical flow) [3]. In this project, we implement an algorithm for 3D motion and structure reconstruction using optical flow field from sequence of images of moving objects. More specifically, using the optical flow field and intrinsic parameters of the camera, we recover the 3D motion and structure of the moving object with respect to camera reference frame [4].

2 Theory

Optical flow is an estimate of the projection of the image motion field along the direction of image gradients. Given the estimate of the optical flow field, denoted as $[v_x, v_y]^T$, the goal is to determine the 3D translational and rotational motion, denoted by $[T_x, T_y, T_z]^T$ and $[\omega_x, \omega_y, \omega_z]^T$ respectively. Furthermore, we want to estimate the 3D coordinates of the moving object $[X, Y, Z]^T$ from the optical flow field.

The estimation process can be divided into three major Steps:

Step 1: Determination of the translational motion, $[T_x, T_y, T_z]^T$

Step 2: Determination of the rotational motion, $[\omega_x, \omega_y, \omega_z]^T$

Step 3: Determination of the depth, Z

Let, $p = [x, y]^T$ and $\bar{p} = [\bar{x}, \bar{y}]^T$ are very near to each other in an image frame (this property of two image points is referred to as *approximately instantaneously coincident points*). Let, the corresponding 3D points as $P = [X, Y, Z]^T$ and $\bar{P} = [\bar{X}, \bar{Y}, \bar{Z}]^T$. Then, the relative motion field, $[\Delta v_x, \Delta v_y]^T$ between the two points can be decomposed into the translational component ($[\Delta v_x^T, \Delta v_y^T]^T$) and the rotational component ($[\Delta v_x^\omega, \Delta v_y^\omega]^T$).

It can be shown that,

$$\begin{aligned}\Delta v_x^T &= (T_z x - T_x f)(\frac{1}{Z} - \frac{1}{\bar{Z}}) + \frac{T_z}{Z}(x - \bar{x}) \\ \Delta v_y^T &= (T_z y - T_y f)(\frac{1}{Z} - \frac{1}{\bar{Z}}) + \frac{T_z}{Z}(y - \bar{y})\end{aligned} \tag{1}$$

and,

$$\begin{aligned}\Delta v_x^\omega &= \omega_z(y - \bar{y}) + \frac{\omega_x}{f}(xy - \bar{x}\bar{y}) - \frac{\omega_y}{f}(x^2 - \bar{x}^2) \\ \Delta v_x^\omega &= -\omega_z(x - \bar{x}) - \frac{\omega_y}{f}(xy - \bar{x}\bar{y}) + \frac{\omega_x}{f}(y^2 - \bar{y}^2)\end{aligned} \tag{2}$$

where, $[T_x, T_y, T_z]^T$ ($(\omega_x, \omega_y, \omega_z)^T$) is the 3D translational (rotational) motion field of the point p which we want to estimate.

As p and $\bar{p}$ are very near in the image frame, $\Delta v_x^\omega = \Delta v_y^\omega = 0$. Now, if the corresponding 3D points P and $\bar{P}$ are far from each other, $|Z - \bar{Z}| \gg 0$ which implies Δv_x^T and Δv_y^T are large. In this case, only the translational component can describe the relative motion. This property is referred to as **"Motion Parallax"**.

It can be shown that, motion parallax at p can be estimated by the optical flow difference between p and $\bar{p}$. Moreover, we know that, for any two instantaneously coincident points (in our case, p and $\bar{p}$ are approximately satisfying the condition), the relative motion $[\Delta v_x, \Delta v_y]^T$ is directed towards the instantaneous epipole ($p_0 = [x_0, y_0]^T$).

Now, given the relative motion field $\left([\Delta v_{x_i}, \Delta v_{y_i}]^T\right)_{i=1}^N$ of N points $(p_1, ..., p_N)$ which satisfying the motion parallax, we can solve a least square problem which indeed can be used to estimate T_x, T_y upto T_z as follows:

$$\begin{aligned}T_x &= x_0 \frac{T_z}{f} \\ T_y &= y_0 \frac{T_z}{f}\end{aligned} \tag{3}$$

Estimating p_0 from the given N points.

Given the N points well approximating the motion parallax, we form the matrix A_i for the point p_i $(i = 1, 2, ..., N)$ as,

$$A_i = \begin{bmatrix} \sum \Delta^2 v_x & \sum \Delta v_x \Delta v_y \\ \sum \Delta v_x \Delta v_y & \Delta^2 v_y \end{bmatrix}$$

where the sum is over the neighbors of p_i in a patch Q_i. Δv_x and Δv_y are estimated as mentioned above by the optical flow difference over the neighbors.

We estimate the motion parallax of p_i (i.e. the direction of the instantaneous epipole p_0 from p_i) denoted as $\hat{I}_i$ as the eigenvectors of A_i with the largest eigenvalue. Now, as $\hat{I}_i$, p_0 and p_i are co-planar, we can write

$$(\hat{I}_i \times p_i)^T p_0 = 0$$

Therefore, for the N points with N patches, we can formulate the matrix B as,

$$B = \begin{bmatrix} \hat{I}_1 \times p_1^T \\ \hat{I}_2 \times p_2^T \\ . \\ . \\ . \\ \hat{I}_N \times p_N^T \end{bmatrix}, \text{ with } Bp_0 = 0$$

Then, the problem is to determine p_0 as the least-square estimate that satisfies $Bp_0 = 0$. The solution can be found by decomposing WB by SVD such that $WB = UDV^T$. The column of V corresponding to the smallest singular value will represent p_0 (here, W is a diagonal matrix in which the diagonal entries are the largest eigenvalues of A_i).

Once, we estimate the epipole p_0, we can estimate the rotational 3D motion $[\omega_x, \omega_y, \omega_z]^T$. Let, $v_\perp$ is the inner product between the optical flow $[v_x, v_y]^T$ and the vector $[y_i - y_0, -(x_i - x_0)]^T$ of the point $p_i = [x_i, y_i]^T$ for $i = 1, 2, ..., N$. Then,

$$v_\perp = v_x^\omega (y_i - y_0) - v_y^\omega (x_i - x_0) \tag{4}$$

Here,

$$v_x^\omega = -\omega_y f + \omega_z y + \frac{\omega_x xy}{f} - \omega_y \frac{x^2}{f}$$
$$v_y^\omega = \omega_x f - \omega_z x - \frac{\omega_y xy}{f} + \frac{\omega_x y^2}{f}$$

are the rotational component of $[v_x, v_y]^T$. Therefore, we can estimate $[\omega_x, \omega_y, \omega_z]^T$ by solving the least square problem of the system of linear equations formed by the equation (4) with $i = 1, 2, .., N$.

Now, as

$$\begin{aligned} v_x &= \frac{T_z x - T_x f}{Z} - \omega_y f + \omega_z y + \frac{\omega_x xy}{f} - \frac{\omega_y x^2}{f} \\ v_y &= \frac{T_z y - T_y f}{Z} + \omega_x f - \omega_z x - \frac{\omega_y xy}{f} + \frac{\omega_x y^2}{f} \end{aligned} \tag{5}$$

replacing T_x, T_y from equation (3) and using the estimate of $[\omega_x, \omega_y, \omega_z]^T$ we can solve for T_z ad Z and hence T_x, T_y using equation (3).

Now given the intrinsic camera parameters i.e. the perspective projection matrix $P = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_34 \end{bmatrix}$ we can solve for X and Y using the weak perspective projection i.e. by solving the following two equations:

$$p_{11}X + p_{12}Y + p_{13}Z + p_{14} = xZ$$
$$p_{21}X + p_{22}Y + p_{23}Z + p_{24} = yZ$$

A summary of the **algorithm** has been given below:

Step 1: Calibrate the camera using a calibration pattern to recover the matrix P and the parameter f (i.e. the intrinsic camera parameter)(as in the Class project 2 of the course)

Step 2: Obtain the optical flow of the image of interest (i.e. for which we want to estimate the 3D motion) within the sequence of images captured by the camera calibrated at Step 1 (as in the Home Work 6 of the course)

Step 3: Write equation (5) in the image reference frame, using the knowledge of the intrinsic parameters

Step 4: For each image point p_i, $i = 1, 2, ..., N$

- Compute the flow difference Δv_x and Δv_y between the optical flow at p_i and at all the points p in a neighborhood of p_i, Q_i
- Compute the eigenvalues and the eigenvectors of the matrix A_i; let λ_i be the greater eigenvalue, and $\hat{I}_i$ the unit eigenvector corresponding to λ_i

Step 5: Compute the SVD of WB, $WB = UDV^T$. Estimate the epipole of p_0 as the column of V corresponding to the smallest singular value

Step 6: Form the dot product of equation (4) for $i = 1, 2, ..., N$ and rewrite the equations obtained in the image reference frame

Step 7: Determine the angular velocity components as the least-square solution of a system of N simultaneous instances of equation (4)

Step 8: Determine the translational direction from the epipole coordinates and the knowledge of intrinsic parameters

Step 9: Solve equation (5) for the depth of each image points

The output quantities are the direction of translation, angular velocity and the 3-D coordinates of the scene points

3 Data Collection

3.1 Data collection for camera calibration

- We used a high-contrast calibration pattern, a chessboard, and printed it on a flat hardboard.
- We made sure the camera was mounted in a fixed position and pointed directly to the pattern.
- We then captured multiple images of the chessboard from different angles and distances. We also varied the orientation and position of the chessboard within the camera's field of view.

The whole set of calibration images is uploaded to our box folder Some of the sample images used for calibration are shown in Fig. 1.

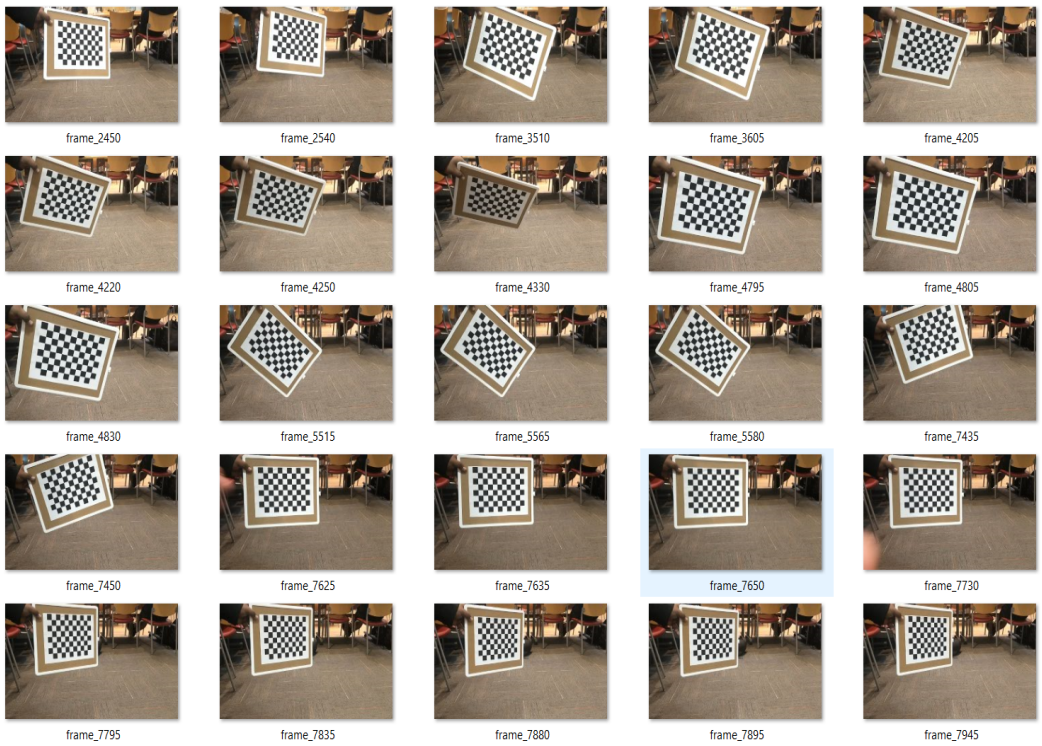


Figure 1: Sample images for camera calibration

3.2 Data collection for Optical flow and, Motion and structure detection

- We set up a controlled environment, in an open field, ensuring ample sunlight was available.
- We placed a medium size box on a vertical stand (tripod), and then captured a sequence of images or a video that includes the motion we want to analyze. The box and the background open field had a significant distance between them. We tried our best to have a motion parallax between the box and the far outfield while capturing the sequences of images.

Some of the sample frames of this video are presented in Fig. 2.

4 Experimental Results

4.1 Camera calibration

In this project, we used MATLAB's camera calibration app to find our camera's intrinsic parameters. We believe this app will give us more accurate results than our custom-written code. The camera calibrator app uses Zhang's Calibration Method for camera calibration [5] that utilizes multiple images (We used 30) of a planar pattern viewed from different angles. It also employs iterative techniques to reduce re-projection discrepancies, which helps us find the optimal calibration parameters. The 3×3 intrinsic matrix it generates has the following form:

$$I = \begin{bmatrix} f_x & s & c_x \\ 0 & f_x & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{6}$$

Figure 2: Sample frames for optical flow and, motion and structure detection data

where, the coordinates (c_x, c_y) represent the optical center, and the skew parameter, s equals 0 when the $x-$ and $y-$ axes are exactly perpendicular [5]. In the above matrix, $f_x = F * s_x$, and $f_y = F * s_y$. Here, F is the focal length, usually in mm, and s_x and s_y are the number of pixels per world unit in their respective direction.
After calibrating our camera, we found the following intrinsic matrix:

$$I = \begin{bmatrix} 1979.39 & 0 & 958 \\ 0 & 1979.39 & 517 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

where, the focal length, $F = 1.97939$ mm, and $(c_x, c_y) = (958, 517)$. s_x and s_y are both found to be 1000.

4.2 Optical flow computation

We implement the Lucas-Kanade method for calculating the optical flow of the frames[1]. As shown in Fig. 3 (for frame 50), the flow estimation is reasonably accurate for different objects in the scene.

4.3 Structure reconstruction

We present the dense reconstructed structure of Object 1 (box) in Fig. 4 (estimated from frame 50). As we can observe from the figure, the reconstruction error is much higher on the vertical sides compared to the horizontal sides as the motion parallax holds more accurately along the horizontal

[1]We implement the MATLAB function for the Lucas-Kanade method for precise calculation of the flow as the task is not the scope of the project [6]

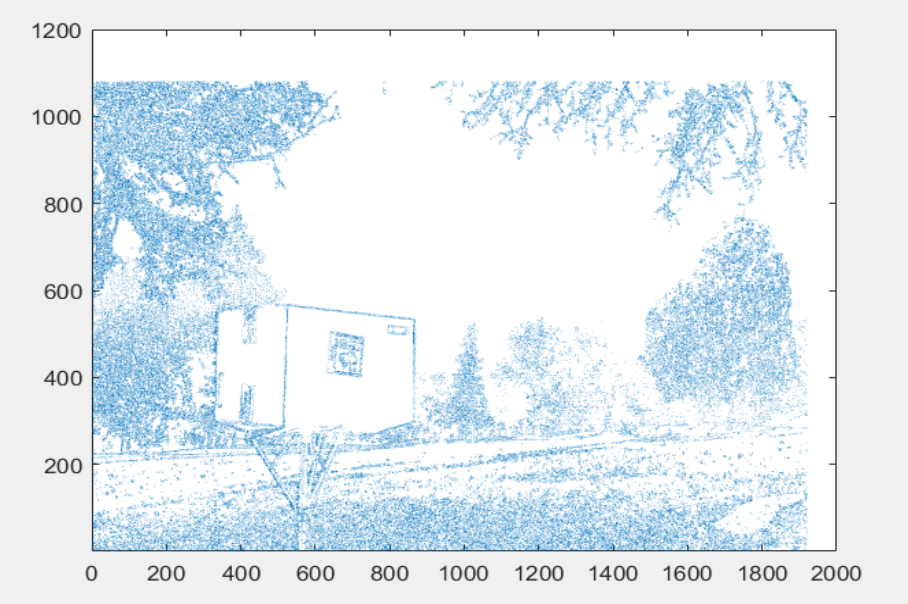


Figure 3: Optical flow for frame 50

sides compared to the vertical sides. This represents a high dependency of the algorithm on motion parallax for estimating the relative image motion field using the flow difference.

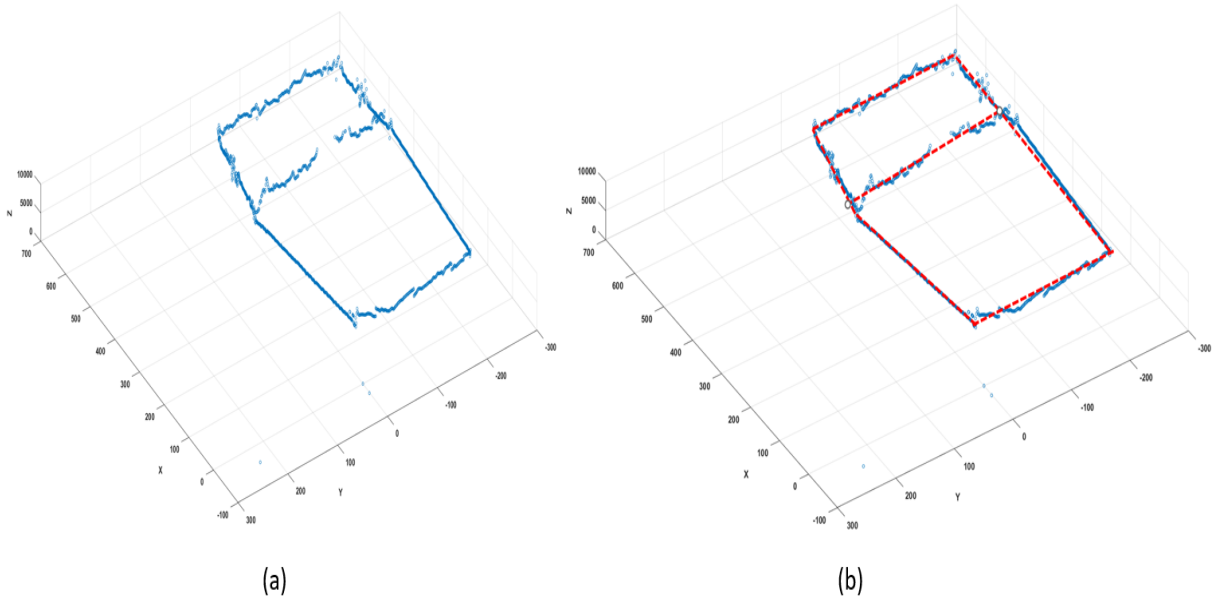
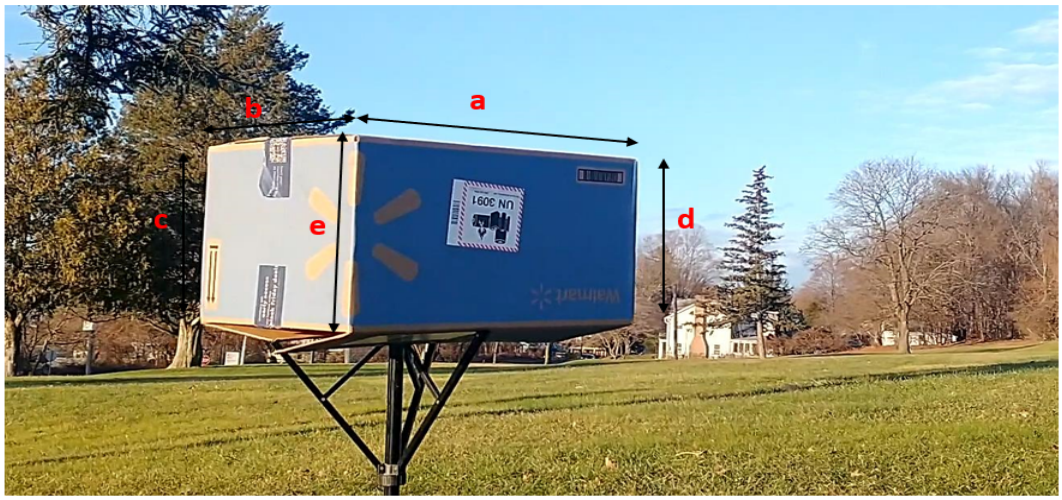


Figure 4: (a) 3D structure reconstruction (dense) of Object1 (box), (b) Approximate reconstructed shape

4.4 Dimension measurement

From the reconstructed structure, we calculated the dimensions of various sides of the object. Again, as we can see from Fig. 5, the measurement is more accurate for the sides where the assumption of motion parallax holds more strongly than others.



Sides	Actual length (mm)	Measured length (mm)
a	584.2	880.31
b	431.8	557.42
c	317.14	644.84
d	317.14	452.62
e	317.14	275.36

Figure 5: (a) Different sides of Object 1 and their measured dimensions

4.5 Estimation of depth

We estimate the depth of different objects in the scene. As shown in Fig 6, Table 1 and from our perception, the depth estimation is more accurate for Object 1 compared to Object 2 and Object 3 as motion parallax approximation occurs more strongly for the former compared to the latter two objects.

Objects	Depth (mm)
Object 1 (box)	2197.64
Object 2 (tree1)	2811.92
Object 3 (tree2)	4338.27

Table 1: Eastimated depth of objects

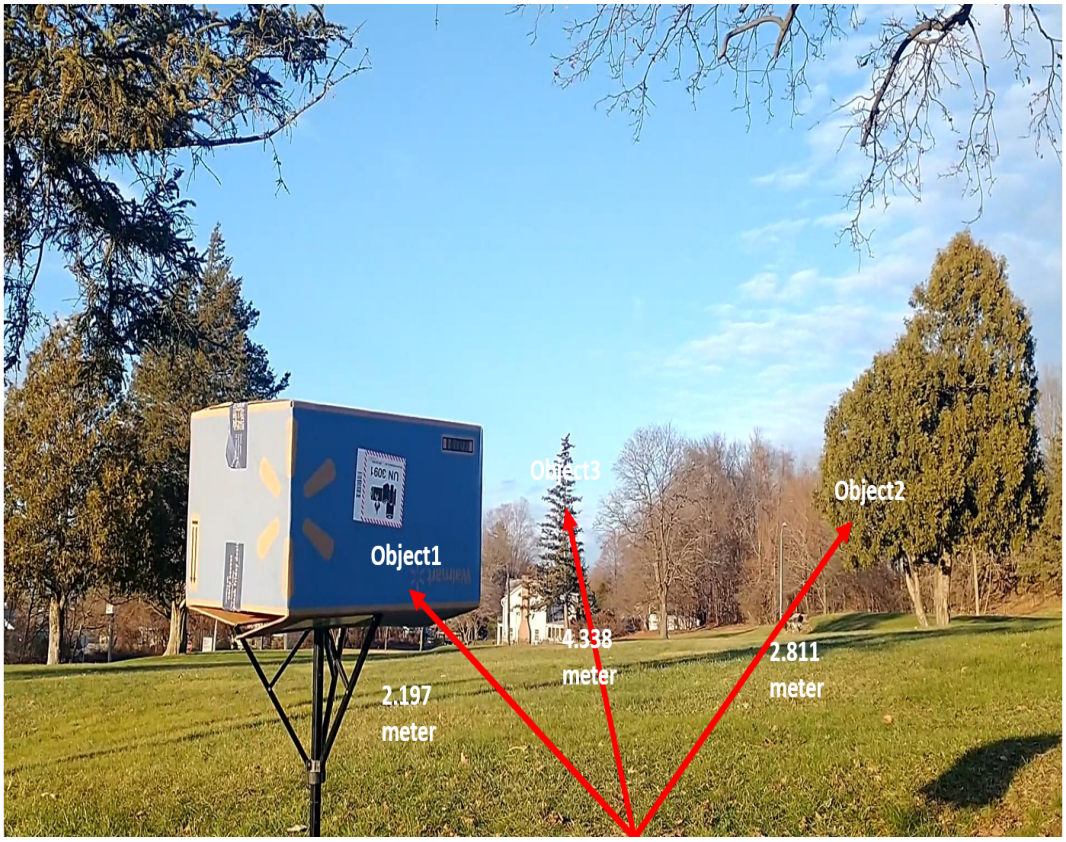


Figure 6: Estimated depth of the selected objects

4.6 Estimation of motion

Finally, we estimate the motion i.e. the 3D velocity of different objects (Object 1, Object 2, and Object 3) of the scene from the estimated translational and rotation 3D motion field. We plot the estimated 3D velocity of the objects in Fig. 7, 8, and 9 respectively. As we can see and also mentioned earlier, the motion estimation is also more accurate (please see the attached video for validation) due to the stronger motion parallax in Object 1 compared to the other two objects.

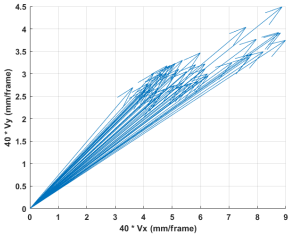


Figure 7: The 3D velocity of the Object 1 (Z-axis is inward perpendicular to the page/screen

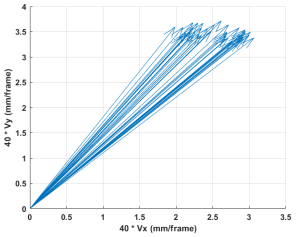


Figure 8: The 3D velocity of the Object 2 (Z-axis is inward perpendicular to the page/screen

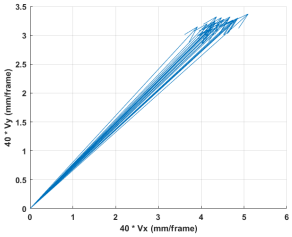


Figure 9: The 3D velocity of the Object 2 (Z-axis is inward perpendicular to the page/screen

5 Task Distribution

The group members contributed equally to the implementation of the project. Different tasks were distributed among the group members as given in Table 2.

Mohammed Nowaz Rabbani Chowdhury	**Sudipta Paul**
Data collection (camera calibration)	Data collection (OF/Motion and structure detection)
Implementation of the camera calibration algorithm	Implementation of the algorithm for optical flow estimation
Implementation of the algorithm for 3D Motion and structure estimation (joint work)	Implementation of the algorithm for 3D Motion and structure estimation (joint work)
Measurement of object dimension, directional velocity plot	Depth estimation of selected objects
Report writing, presentation (joint work)	Report writing, presentation (joint work)

Table 2: Work distribution

6 Conclusion

In this project, we performed 3D motion and structure reconstruction from the optical flow of the sequence of images. Before jumping into the main task, we performed camera calibration and optical flow estimation using MATLAB's built-in toolboxes/apps. It is worth mentioning that optical flow, even though, provides a dense estimate of motion field, often these estimates are inaccurate. However, if strong motion parallax holds, then the optical flow provides a good estimate of the motion field of the moving objects. After computing camera calibrations and optical flow, the obtained results are used to implement the algorithm of motion and structure reconstruction. In addition to reconstructing 3D objects and computing 3D velocity, we performed dimension measurement and depth estimation of some selected objects. Our analysis shows that for precise motion estimation and object reconstruction, a strong motion parallax is required in this method.

References

[1] Stefano Soatto, Ruggero Frezza, and Pietro Perona. Motion estimation via dynamic vision. *IEEE Transactions on Automatic Control*, 41(3):393–413, 1996.

[2] Tony Jebara, Ali Azarbayejani, and Alex Pentland. 3d structure from 2d motion. *IEEE Signal processing magazine*, 16(3):66–84, 1999.

[3] Emanuele Trucco and Alessandro Verri. *Introductory techniques for 3-D computer vision*, volume 201. Prentice Hall Englewood Cliffs, 1998.

[4] David J Heeger and Allan D Jepson. Subspace methods for recovering rigid motion i: Algorithm and implementation. *International Journal of Computer Vision*, 7:95–117, 1992.

[5] MATLAB. cameraparameters. https://www.mathworks.com/help/vision/ref/cameraparameters.html.

[6] MATLAB. Lucas-kanade. https://www.mathworks.com/help/vision/ref/opticalflowlk.html.