# 3D Localisation of Objects Using Multiple Cameras

Garima Singh, *Student*, Geereddy Sathwika Reddy, *Student,* and Ishwar Lal Kumawat, *Student,* under Dr. Sandip Ghosh, *Associate Professor,* Department of Electrical Engineering, Indian Institute of Technology (BHU), Varanasi.

***Abstract*** **- This report presents a study on the 3D localization of objects using multiple cameras. The study aimed to develop a robust and accurate method for localizing objects in 3D space using a network of synchronized cameras. The approach involved camera calibration, feature extraction and matching, triangulation, and optimization. The study makes a novel contribution to computer vision by providing a robust and efficient approach to 3D object localization, with potential applications in fields such as robotics, surveillance, and augmented reality.**

## 1. Introduction

3D localization is needed in many applications where it is essential to accurately determine the position and orientation of objects in three-dimensional space. In robotics, 3D localization is crucial for object recognition, grasping, and manipulation tasks. Surveillance can track the movement of people or objects in 3D space, providing more detailed information about their behavior and interactions. 3D localization is essential in virtual reality to create a convincing and immersive virtual environment. It is also useful in localizing electrical faults in power transmission lines, tracking drones or other aerial vehicles, and mapping indoor environments for energy-efficient building design.

This paper proposes a method for 3D object localization and tracking using multiple cameras. The method works by first detecting and tracking the object in each camera view and then using the corresponding points in multiple views to reconstruct the 3D position of the object. The method consists of two main steps: camera calibration and object tracking.

Each camera's intrinsic and extrinsic parameters are determined using a calibration pattern in the camera calibration step. This step is necessary to estimate the 3D position of the object accurately. In the object tracking step, the object is detected and tracked using color and shape features in each camera view. After this, coordinates are transformed from one plane to another.

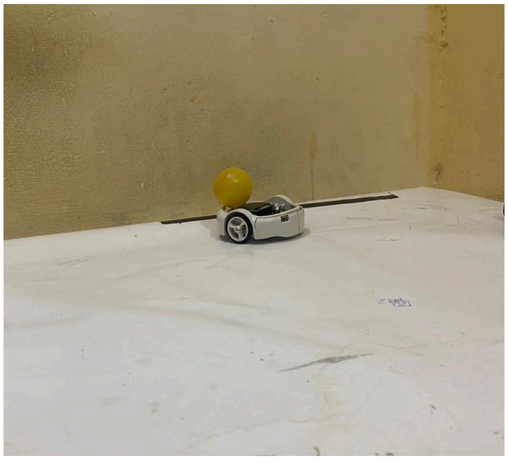

Fig 1. Setup of a yellow ball in the plane

## 2. Camera Calibration

### A. Intrinsic Parameters

The intrinsic properties of a camera are fixed and defined by the manufacturer. These properties include the focal length, sensor size, pixel size, principal point, and lens distortion parameters. They are related to the internal geometry of the camera and are independent of the position and orientation of the camera in the world coordinate system.

The focal length determines the camera's magnification and is usually given in millimeters. The sensor size and pixel size determine the camera's field of view and resolution. The principal point is the point on the image sensor where the camera's optical axis intersects and is usually given in pixel coordinates. The lens distortion parameters account for imperfections in the lens that can cause image distortion, such as radial distortion, tangential distortion, and other types of distortion. The intrinsic parameters of the cameras used are as follows:

$$fx1 = 595.4996;\ fy1 = 596.40136$$
$$cx1 = 310.64632;\ cy1 = 236.42889$$

```
intrinsic_matrix_cam1 = np.array(
          [[fx1, 0, cx1, 0]
           [0, fy1, cy1, 0]
           [0,    0,     1, 0]
           [0,    0,     0, 1]]  )
```

distortion_coefficients: rows: 1; cols: 5;
data:[-0.419815 0.170359 0.002695 0.000189 0.000000]

### B. Extrinsic Parameters

The extrinsic parameters of a camera refer to its position and orientation in the world coordinate system. The extrinsic parameters include:

Rotation matrix (R): This matrix describes the camera's orientation with respect to the world coordinate system. It is a 3x3 matrix that defines the transformation of the camera coordinate system to the world coordinate system. We can define the matrix by first defining a unit vector that represents the axis of rotation and then applying a rotation about this axis by an angle θ. The resulting rotation matrix will be:

R = [cos(θ) + u_x^2(1-cos(θ))    u_x*u_y*(1-cos(θ)) - u_z*sin(θ)    u_x*u_z*(1-cos(θ)) + u_y*sin(θ)]
[u_y*u_x*(1-cos(θ)) + u_z*sin(θ)    cos(θ) + u_y^2(1-cos(θ))    u_y*u_z*(1-cos(θ)) - u_x*sin(θ)]
[u_z*u_x*(1-cos(θ)) - u_y*sin(θ)    u_z*u_y*(1-cos(θ)) + u_x*sin(θ)    cos(θ) + u_z^2(1-cos(θ))]

where u = (u_x, u_y, u_z) is the unit vector representing the rotation axis, and cos(θ) and sin(θ) are the cosine and sine of the rotation angle, respectively.

Translation vector (t): This vector describes the camera's position with respect to the world coordinate system. It is a 3x1 vector that defines the translation of the camera coordinate system to the world coordinate system. If the position of the camera in the world coordinate system is given by P_cw and the position of the camera coordinate system's origin in the world coordinate system is given by O_cw, then the translation vector t is given by:

t = P_cw - O_cw

where P_cw = [X_cw, Y_cw, Z_cw]$^T$ and O_cw = [X_ocw, Y_ocw, Z_ocw]$^T$

Together, the rotation matrix and translation vector define the transformation matrix that maps points from the camera coordinate system to the world coordinate system. This matrix is typically denoted as [R | t], where "|" represents matrix concatenation.

$$s\begin{bmatrix}u\\v\\1\end{bmatrix}=\begin{bmatrix}f_x & 0 & c_x\\0 & f_y & c_y\\0 & 0 & 1\end{bmatrix}\begin{bmatrix}r_{11} & r_{12} & r_{13} & t_1\\r_{21} & r_{22} & r_{23} & t_2\\r_{31} & r_{32} & r_{33} & t_3\end{bmatrix}\begin{bmatrix}X\\Y\\Z\\1\end{bmatrix}$$

2D Image Coordinates    Intrinsic properties (Optical Centre, scaling)    Extrinsic properties (Camera Rotation and translation)    3D World Coordinates

Fig 2. Camera projection matrix

### C. Robot Operating System (ROS)

The implementation of camera calibration using the Robot Operating System (ROS) is outlined in this project report. Leveraging ROS as a versatile open-source middleware framework facilitated seamless integration and communication among diverse robotic components. The calibration process focused on refining intrinsic and extrinsic camera parameters to enhance the precision of spatial perception. Employing ROS provided a systematic and efficient calibration procedure, ensuring optimal alignment between the physical and virtual realms. This calibration procedure, outlined in the report, is poised to contribute significantly to improved perception accuracy within robotic systems, thereby establishing a foundation for reliable and robust operations across various applications, including autonomous navigation, object recognition, and manipulation.

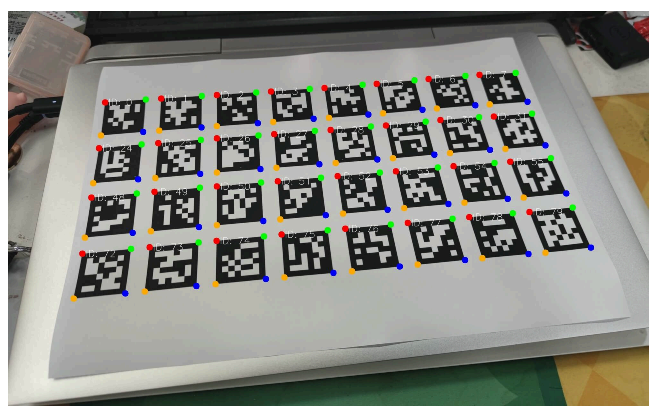

Fig 3. Aruco board model

The camera calibration procedure through ROS was executed in a systematic manner. Initial data acquisition involved capturing a diverse set of images using the calibrated camera setup, ensuring coverage of various perspectives and depths for robust parameter estimation. Leveraging the ROS camera calibration package, specifically "camera_calibration," intrinsic and extrinsic calibration was performed using a chessboard pattern as a target. The acquired images and corresponding calibration data were processed through ROS tools, yielding crucial intrinsic parameters (e.g., focal length, distortion coefficients) and extrinsic parameters (translation and rotation). The iterative adjustment of calibration parameters aimed to minimize errors and achieve optimal alignment between the virtual and physical camera models. The calibration results were thoroughly validated using ROS visualization tools, ensuring the accuracy of the calibrated camera system. This calibrated setup, seamlessly integrated within the ROS ecosystem, establishes a foundation for precise spatial perception in diverse robotic applications.

## 3. Object Detection

Object tracking is locating and following a specific object or multiple objects in a video stream over time. Object tracking aims to identify and track the movement of an object in a scene, even as it changes position, orientation, size, and appearance. Object tracking has many applications, such as surveillance and security, video analysis, robotics, and self-driving cars.

Object detection is the process of locating instances of objects in an image or video and classifying them into predefined categories. The first step is acquiring the image or video stream using a camera or other suitable imaging device. The acquired images or video frames may need preprocessing to improve the image's quality or extract useful information from the image. This may involve color normalization, image enhancement, noise reduction, or image resizing techniques. Once the preprocessing is done, relevant features are extracted from the image. The extracted features should be discriminative enough to distinguish the object of interest from other objects in the image. Now, the image regions likely to contain the object of interest are identified using object proposal techniques. Once the object proposals are generated, the next step is to classify the object of interest into predefined categories.

There are many libraries available that provide pre-trained models for object detection, such as OpenCV, TensorFlow, and PyTorch. OpenCV provides several built-in object detection algorithms that can be used to detect objects in images or videos.

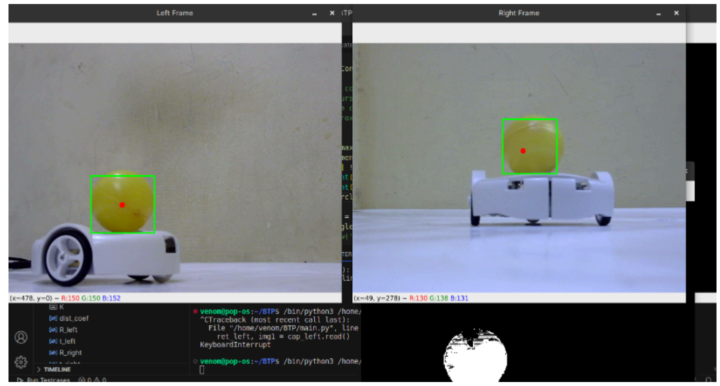

Fig 4. Object Centroid Detection

## 4. Object Tracking

### A. Plane Conversion

Different sensors, cameras, or devices may use different coordinate systems, so it is necessary to convert the coordinates to a common reference frame to perform 3D localization accurately. We need to convert the 2D image coordinates of the point in the camera frame to homogeneous coordinates. We then use the inverse of the camera calibration matrix to convert the homogeneous coordinates from the pixel frame to the normalized camera frame. The camera calibration matrix includes the camera's intrinsic parameters, such as the focal length and principal point. After that, we use the camera's extrinsic parameters to perform a coordinate transformation from the camera frame to the world frame. This can be done by multiplying the normalized camera coordinates by the rotation matrix and adding the translation vector. Finally, we convert the resulting 3D homogeneous coordinate to 3D Cartesian coordinates by dividing by the last coordinate.

Once we have the transformation matrix, we can use it to transform a point P from world coordinates to camera coordinates using the following equation:

$$Pc = T * Pw$$

where Pc is the point in camera coordinates, Pw is the point in world coordinates, and T is the 4x4 transformation matrix that describes the position and orientation of the camera.

To perform this transformation, we need to represent the points in homogeneous coordinates by appending a 1 to their coordinates, as follows:

$$Pc = [Xc, Yc, Zc, 1]T$$

$$Pw = [Xw, Yw, Zw, 1]T$$

where Xc, Yc, Zc, Xw, Yw, and Zw are the coordinates of the points in the camera and world coordinates, respectively.

We then multiply the transformation matrix T by the point in world coordinates Pw, as follows:

[T11 T12 T13 T14] [Xw] =[Xc]
[T21 T22 T23 T24] [Yw] = [Yc]
[T31 T32 T33 T34] [Zw] =[Zc]
[ 0 0 0 1 ] [ 1] =[ 1]

The resulting vector [Xc, Yc, Zc, 1]T is the point in camera coordinates. To convert it back to 3D coordinates, we divide the first three components by the fourth component, as follows:

$$X = Xc / Zc$$
$$Y = Yc / Zc$$
$$Z = 1$$

The resulting vector [X, Y, Z]T is the point in camera coordinates in 3D space.

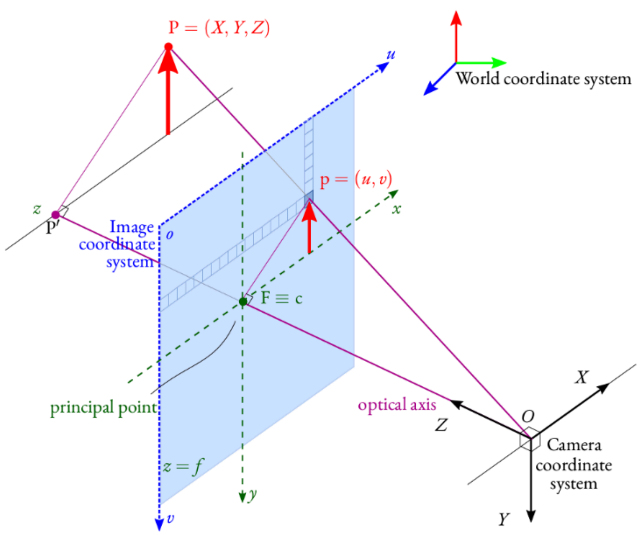


Fig 5. Camera coordinate and World coordinate systems

## B. Triangulation

Triangulation is the process of determining the 3D position of a point in space by measuring its projections onto two or more imaging sensors. In 3D localization, triangulation is used to determine the precise location of an object by combining the 2D coordinates of the object in multiple images obtained from different cameras.

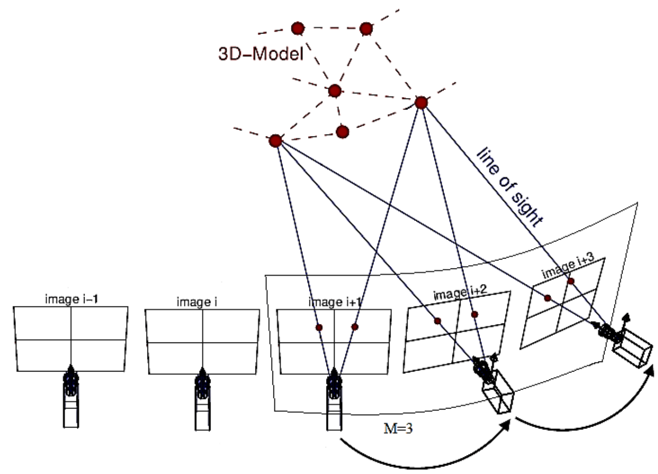


Fig 6. Reconstruction of 3D points using triangulation

The basic principle behind triangulation is to find the intersection point of multiple lines of sight from different viewpoints. By calculating the intersection point of the lines of sight, we can obtain the 3D position of the object. Firstly, we compute the projection matrices of the cameras. The general form of the projection matrix is given as:

$$P = K[R|t]$$

where P is the 3x4 projection matrix, K is the 3x3 intrinsic matrix, R is the 3x3 rotation matrix representing the orientation of the camera, and t is the 3x1 translation vector representing the position of the camera.

For each pair of cameras that observe the same point of interest, compute the two rays that pass through the corresponding 2D image points and emanate from the camera centers in the direction of the points. We then compute the intersection of the two rays using linear algebra. This intersection point corresponds to the 3D location of the point of interest in the world coordinate system. We repeat it for all pairs of cameras that observe the same point of interest.

## 5. Results Interpretation

The applied algorithm was employed for a dual-camera system, and the individual perspectives from each camera are presented herein. Additionally, the reconstructed final localization trace of the object as captured by both cameras, has been successfully obtained.

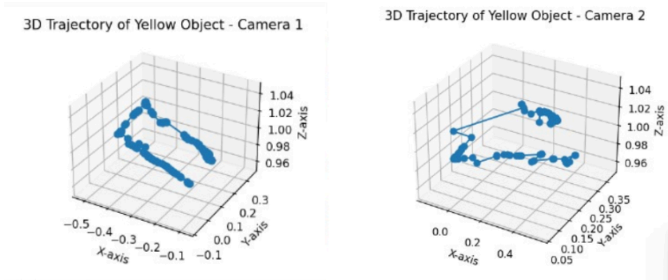


Fig 7. 3D Trajectory from individual cameras

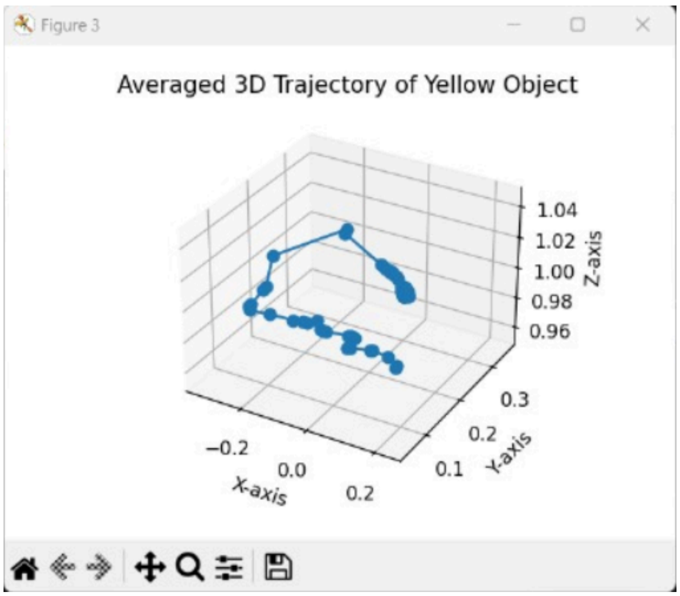


Fig 8. Final 3D localization trace of the object

## 7. References

[1] T. Svoboda and J. Matas, “Multi-camera tracking and 3D reconstruction of rigid objects”
[2] G. Gallego and A. Gil, “A vision-based multi-camera system for 3D localization of vehicles”
[3] S. Wang, J. Shen, and Y. Wang, “Multi-camera 3D object detection and tracking in warehouse environments”
[4] A. Fetić, D. Jurić and D. Osmanković, "The procedure of a camera calibration using Camera Calibration Toolbox for MATLAB,"
[5] J. Chen, D. Wu, P. Song, F. Deng, Y. He and S. Pang, "Multi-View Triangulation: Systematic Comparison and an Improved Method,"