Robotics Project Camera to Real-World Co-ordinates

CSE 432 Robotics

Ahmed Anwar Mazhar Abdelmoamen Gad

ID : 120210007

Assignment submitted to

Dr. Ossama Ismail

Computer Science Engineering Department

Egypt-Japan University of Science and Technology

Egypt

1 Introduction

The objective of this project is to leverage computer vision techniques, specifically using a camera matrix, to obtain real-world coordinates from image data and enable navigation in a simulated environment. The approach revolves around camera calibration, the detection of markers in the camera frame, and the conversion of pixel coordinates into real-world coordinates. These real-world coordinates are then used to navigate a robot in a simulation.

2 Concepts and Methods

2.1 Camera Calibration

Camera calibration is the process of determining the intrinsic and extrinsic parameters of the camera. These parameters allow us to map pixel coordinates in an image to real-world coordinates in a 3D space.

2.1.1 Intrinsic Parameters

These describe the internal characteristics of the camera, such as the focal length and the optical center (principal point). They are stored in the **camera matrix**. The **camera matrix** is a 3x3 matrix that contains the intrinsic parameters:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

Where:

- f_x and f_y are the focal lengths of the camera in the x and y directions (measured in pixels),
- c_x and c_y are the coordinates of the optical center (the principal point) in the image.

2.1.2 Extrinsic Parameters

Extrinsic parameters describe the position and orientation of the camera in the world, which define the transformation between the world coordinate system and the camera coordinate system. These parameters are essential when the camera is not fixed in space and we need to understand its pose (position and orientation) relative to the world.

The extrinsic parameters are typically represented by a rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$. These parameters together form the transformation matrix $\mathbf{T}$ that maps a point $\mathbf{P}_{world} = (X, Y, Z)$ in world coordinates to the corresponding point $\mathbf{P}_{camera} = (x_c, y_c, z_c)$ in the camera coordinate system:

$$\mathbf{P}_{camera} = \mathbf{R} \cdot \mathbf{P}_{world} + \mathbf{t}$$

Where: - $\mathbf{R}$ is the 3x3 rotation matrix that describes the orientation of the camera relative to the world coordinate system. It rotates the world coordinates to align them with the camera's coordinate system. - $\mathbf{t}$ is the 3x1 translation vector that describes the position of the camera in the world coordinate system. It translates the world coordinates to the camera's position.

The full transformation from world coordinates $\mathbf{P}_{world} = (X, Y, Z)$ to camera coordinates $\mathbf{P}_{camera} = (x_c, y_c, z_c)$ can be written as:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

Where:

$$\mathbf{R} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}$$

is the rotation matrix, and $\mathbf{t} = [t_x, t_y, t_z]^T$ is the translation vector.

In cases where the camera is fixed and its extrinsic parameters do not change (such as in our simplified case), the extrinsic matrix $\mathbf{R}$ and translation vector $\mathbf{t}$ can be assumed to be constant. As a result, they do not directly affect the camera calibration

process. In practical applications, however, the extrinsic parameters would be crucial for determining the relative positions of objects or the camera's perspective in dynamic environments.

In summary: - Rotation Matrix $\mathbf{R}$: Describes the camera's orientation in the world. - Translation Vector $\mathbf{t}$: Describes the camera's position in the world.

These parameters allow us to transform world coordinates into camera coordinates, but they are not needed for our particular case since the camera is assumed to be fixed and its pose does not change.

2.2 Detecting Markers

A marker, such as a chessboard or a blob, is used in the camera's field of view for detection. The camera detects the marker using computer vision techniques. In our case, we are using `SimpleBlobDetector` to identify a marker and extract its pixel coordinates.

The **key points** (marker positions) are detected, and their coordinates are obtained in pixel space.

2.3 Pixel to Real-World Conversion

Once the marker's pixel coordinates are detected, we need to convert these coordinates into real-world coordinates. This process is crucial for navigation and determining the relative position of objects or the robot.

2.3.1 Normalization of Pixel Coordinates

The first step in transforming pixel coordinates into real-world coordinates is normalizing the image coordinates. This is done using the camera matrix. For a pixel at position (u, v), the normalized coordinates (x, y) can be computed as:

$$x = \frac{u - c_x}{f_x}$$

$$y = \frac{v - c_y}{f_y}$$

Where:

- u and v are the pixel coordinates in the image,
- c_x and c_y are the principal point (the optical center),
- f_x and f_y are the focal lengths of the camera in the horizontal and vertical directions, respectively.

2.3.2 Mapping to Real-World Coordinates

To map these normalized coordinates to real-world coordinates, we need a depth (distance) value, d, which represents the distance from the camera to the marker. This is an essential piece of information and could be measured manually or assumed based on the application.

The real-world coordinates (X, Y, Z) can be calculated using the following formulas:

$$X = d \cdot x$$

$$Y = d \cdot y$$

$$Z = d$$

Where:

- d is the distance from the camera to the marker (in meters or any real-world unit).

By applying these formulas, we can convert the 2D pixel coordinates to 3D world coordinates.

2.4 Robot Navigation

Once the real-world coordinates are obtained, the robot can navigate toward the detected marker or target point in a simulated environment.

2.4.1 Movement Directions

The navigation system computes the direction vector between the robot's current position and the target position (real-world coordinates). The direction vector is normalized, and the robot's speed is applied to move it towards the target.

2.4.2 Control Logic

The robot adjusts its movement based on the distance to the target. Once the robot reaches a certain threshold (e.g., 0.05 meters), it stops moving. This behavior is implemented using simple control logic that calculates the distance to the target in each simulation iteration.

3 Used Approach

The workflow of the system can be summarized as follows:

1. **Camera Calibration**: The camera's intrinsic parameters are manually provided or calibrated using a chessboard pattern. The camera matrix, which contains the focal lengths and optical center, is constructed.

2. **Marker Detection**: A marker in the environment (**blob**) is detected using a blob detector. The pixel coordinates of the marker are extracted.

3. **Pixel to Real-World Conversion**: Using the camera matrix, the pixel coordinates of the detected marker are normalized. The normalized coordinates are then scaled by the known distance (depth) to compute the real-world coordinates.

4. **Robot Navigation**: The robot uses these real-world coordinates to navigate toward the target. The robot calculates the direction vector and adjusts its velocity accordingly. It stops when it reaches the target within a predefined threshold.

4 Flowchart