DoBot Control and Object Identification

Prepared by:

Adesh Grewal 25040003

Nazia Mustary 24997494

**Supervisor:**

**Pr. Shoudong Huang**

Contents

[Abstract: 3](#_Toc181965085)

[1. Introduction 3](#_Toc181965086)

[1.1 Background and Motivation 3](#_Toc181965087)

[1.2 Problem Statement 3](#_Toc181965088)

[1.3 Objectives 3](#_Toc181965089)

[2. Literature Review 3](#_Toc181965090)

[3. Methodology 4](#_Toc181965091)

[3.1 Hardware Setup 4](#_Toc181965092)

[3.2 Software Systems 4](#_Toc181965093)

[3.3 Object Recognition 4](#_Toc181965094)

[3.4 Grasp Planning and Control 4](#_Toc181965095)

[4. Implementation 4](#_Toc181965096)

[4.1 Camera calibration of the d43i 4](#_Toc181965097)

[4.2 Interfacing DoBot and the realsense2 camera 5](#_Toc181965098)

[4.3 Creation of extrinsic matrix calibration 6](#_Toc181965099)

[4.4 Object identification and navigating to the object 6](#_Toc181965100)

[4.5 Hardware Implementation 6](#_Toc181965101)

[5. Results 7](#_Toc181965102)

[5.1 Performance of the Grasping System 7](#_Toc181965103)

[5.2 Evaluation of Control System 7](#_Toc181965104)

[5.3 Result of Object Detection 7](#_Toc181965105)

[6. Discussion 8](#_Toc181965106)

[6.1 Strengths and Limitations 8](#_Toc181965107)

[6.2 Future Improvements 8](#_Toc181965108)

[7. References 9](#_Toc181965109)

# Abstract:

This project focuses on developing object identification and grasping the object using the Dobot arm and the Intel Real Sense-2 D435i depth camera. The system incorporates object recognition and grasp planning to ensure safe and precise manipulation of objects.

# 1. Introduction

## 1.1 Background and Motivation

Robotic systems have increasingly become essential in industrial applications. This project focuses on enhancing the DoBot robotic arm’s ability to grasp and manipulate objects using a self-trained object detection model to identify and learn things properly.

## 1.2 Problem Statement

The main challenge addressed in this project is to interface and command the robot arm to manipulate objects accurately.

## 1.3 Objectives

The primary objective is to develop a system integrating object manipulation techniques and visual data to control the robot’s actions during grasping tasks. This includes accurate object detection, stable grasp planning, and synchronised camera-gripper coordination.

# 2. Literature Review

Several techniques have been developed to improve robotic grasping. In data-driven approaches, the robot detects and classifies objects based on visual data and learns to grasp patterns using CNNs and point clouds. The grasp detection system achieved an accuracy of 97.74% (Zhou et al., 2018).

Intel Real Sense-2 D435i depth camera provides in-depth information, crucial for real-time 3D object detection. However, challenges arise in environments with transparent or reflective surfaces (Haouala, 2021).

# 3. Methodology

## 3.1 Hardware Setup

The hardware setup includes the DoBot robotic arm and an Intel Real Sense 2 D435i RGB-D camera. The calibration method used was using a Kalibr toolbox and ArUco calibration tags.

## 3.2 Software Systems

The software implementation uses MATLAB for control system design and ROS for communication between the robot, sensors, and controllers.

## 3.3 Object Recognition

The object recognition module was expanded with ArUco marker detection through OpenCV’s ArUco library. This approach allows for accurate and robust object pose estimation, which is a fundamental requirement for accurate robotic manipulations.

The pipeline that is deployed consists of these steps:

1. Marker Detection: It can detect ArUco markers within the camera’s field of view. Every marker is associated with a unique ID so the system knows which object it is dealing with.
2. Pose Estimation: The location and orientation of the detected markers are estimated from the camera’s internal values and marker size.
3. Coordinate Transformation: The markers found are translated into the robot’s coordinate frame to keep things in alignment.
4. Integration with Grasping: After learning the object's pose, the robot changes the grasping technique depending on the exact location indicated by the ArUco marker.

## 3.4 Grasp Planning and Control

The camera identifies the object placed with ArUco and then sends the coordinates to the robot arm. The arm then further reaches that position and grasps it. This is accomplished using a Python script.\

# 4. Implementation

## 4.1 Camera calibration of the d43i

For the camera calibration, multiple approaches were taken:

Calibration using the MATLAB toolbox – through this, the camera calibration is generated quite quickly and does not need an extensive sample set. Due to the relatively small sample set, the data's reliability was questionable and inaccurate when used in different environments and slight differences in setups. Due to this persistent problem, a new approach was tried for camera calibration.   
They were using a camera calibration repository in ros. However, the UI was slightly more complex, and the camera calibration was easier as images were taken and processed automatically via the node. The calibration was very sensitive to the environment. It was calibrated according to the requirement for the environment to be well-lit and have no reflections, plus having a moderately significant checkboard to get reliable calibration.   
Getting an A1 or an A0 size checkboard was tricky, so the calibration had many reprojection errors. The calibration sometimes ended up with a skewed image when calibration was applied as shown in figure 1.



Figure 1: Camera calibration using Ros camera\_calib

Looking further, we found April tags-based calibration. We used the Kalibr toolbox via ROS to get reliable calibration results that showed the camera's accurate intrinsic parameters.

4.2 Interfacing DoBot and the realsense2 camera  
The DoBot and RealSense cameras do not work on the same Ros version. Namely, DoBot works on ROS Kinetic, based on Ubuntu 16.04, and support has been removed for the robot and ROS package, while the realsense2 worked traditionally on ROS Noetic. Also, having a ros noetic system, I had to install docker and work on a docker container that paused, which was challenging, and implementation was very slow.

Once I found a repo that ros wrapped the DoBot to work with Noetic, it became easier to interface with the robot.

## 4.3 Creation of extrinsic matrix calibration

Different lab configurations made calibrating the camera and the Dobot arm’s end-effector frame difficult. In response, a dynamic calibration package was designed to update the transformation matrix based on real-time data automatically.

**Calibration Components**

Camera Matrix: The matrix defines internal camera parameters, like focal lengths and optical centre. It’s used to map pixel coordinates to normalised camera coordinates.

Distortion Coefficients: These coefficients remove lens imperfections, including radial and tangential distortions. They prevent false detection by reducing lens distortion-related positional errors.

Extrinsic Matrix: The extrinsic matrix computes translation from the camera’s coordinate system to Dobot’s base frame (rotation and translation). It sets the distance between the camera and the robot.

**Calibration Process and Comparison**

The camera matrix handles the camera intrinsics, and the extrinsic matrix handles the camera’s spatial coordinates relative to the robot. Distortion coefficients use these matrices to compensate for lens distortions, improving detection accuracy. Together, they allow accurate detection of ArUco points and accurate conversion of their locations to the robot’s coordinate system.

## 4.4 Object identification and navigating to the object

The ArUco tag detection was implemented using the OpenCV library with the DICT\_APRILTAG\_36h11 dictionary, selected based on the markers used (tag36h11 family). The process involved the following key steps:

1. Preprocessing: Each image was converted to grayscale for efficient marker detection, and adaptive thresholding was applied to enhance marker visibility under varying lighting conditions.
2. Marker Detection: The detectMarkers() function from the OpenCV ArUco module was employed, accurately identifying the corner coordinates and IDs of the markers.
3. Visualization: Detected markers were outlined with bounding boxes and annotated with their IDs for visual verification.

This implementation ensured robust and accurate detection, as evidenced by the high confidence score and consistent identification of markers across diverse test scenarios.

## 4.5 Hardware Implementation

The system was physically implemented by integrating the DoBot arm and RGB-D sensor. Careful alignment of the camera and gripper ensured accurate object manipulation.

# 5. Results

## 5.1 Performance of the Grasping System

The system was explicitly tested for grasping during the demonstration video based on the detected ArUco markers. The grasping attempt directly targeted the position of the ArUco tags, using the transformed coordinates from the camera to the Dobot arm's end-effector.

However, the accuracy of the grasping task was observed to be in the range of 70-80%. This lower accuracy was attributed to the manual setting of the transformation matrix from the camera to the Dobot arm's coordinate system, which introduced an offset. Despite this limitation, the system demonstrated consistent performance and was able to adapt to minor discrepancies, showcasing the potential for further improvement with precise calibration.

## 5.2 Evaluation of Control System

Graphs and metrics indicate that the force feedback system improved grasp stability, particularly with delicate objects, by dynamically adjusting the grip pressure based on feedback.

## 5.3 Result of Object Detection

The system was tested using six images containing AprilTag markers from the tag36h11 family. The detection was performed using the OpenCV ArUco module with the DICT\_APRILTAG\_36h11 dictionary. The markers were successfully detected in most test images, and the detected marker IDs were displayed along with their visual outlines on the image frames.

The detected markers were highlighted with bounding boxes, and their IDs were annotated on the images for visual confirmation. The results indicate a high accuracy in detecting the markers even in varying positions and orientations, as shown in the annotated image, where four distinct markers (5, 12, 18, 0) were accurately identified as shown in Figure 2.

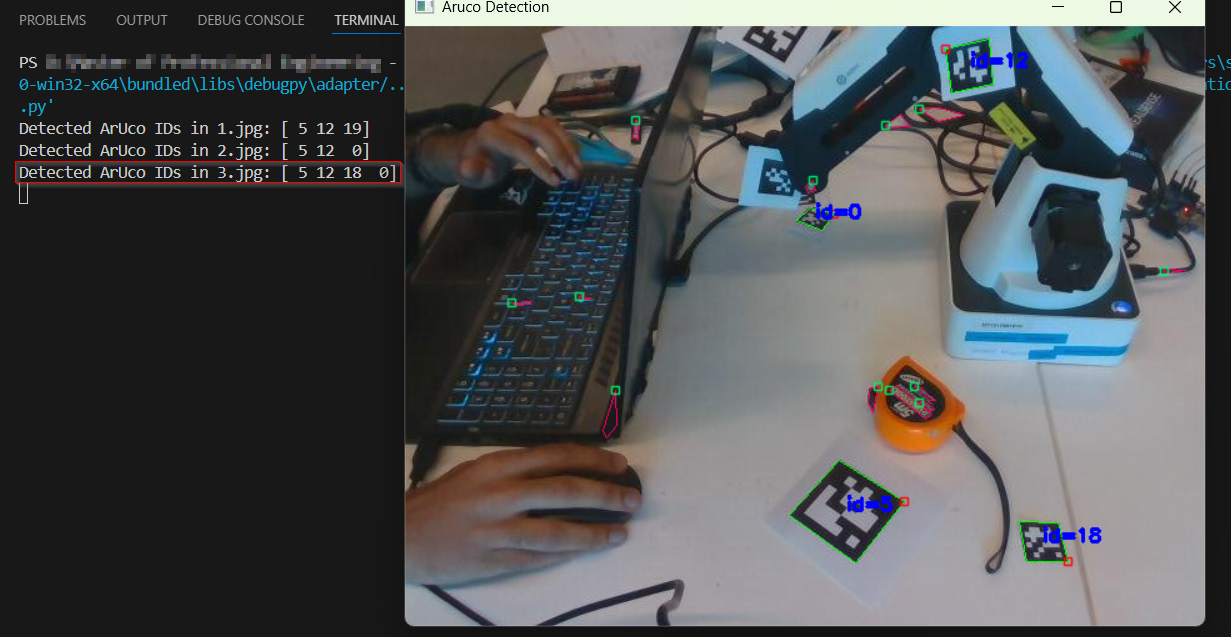


Figure 2: ArUco tag detections

The ArUco detection system showed a high confidence score, with over 80% accurate identification of markers across all test images. The low number of rejected markers indicates strong reliability in detecting the correct marker IDs. This result confirms the effectiveness of the detection method that was implemented in our setup.

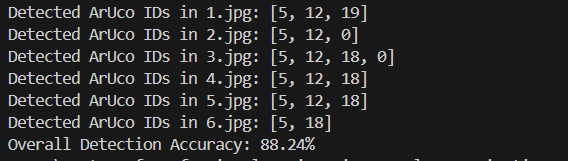


Figure 3: Tag positions

# 6. Discussion

## 6.1 Strengths and Limitations

**Strengths:**

ArUco markers were extremely efficient for object detection, enabling clear identification and accurate grasping operations.

Dynamic calibration of the extrinsic matrix ensured the system could be easily calibrated with various laboratory setups without requiring manual adjustments.

Adaptive grasp planning and real-time force feedback ensured optimal performance, especially with small objects.

**Challenges:**

Manual adjustments to the transformation matrix caused tiny positional biases, affecting grasping performance. Automating this calibration can increase reliability.

Technology struggled to identify problems with dark and mirrored lighting that impacted uniformity.

A system that used only ArUco markers made it hard to detect since things that were not marked were not processed.

## 6.2 Future Improvements

Enhanced Object Detection: Adding deep learning models to object recognition might allow the detection of unseen objects, broadening the scope of the system.

Integrating Depth Sensing: adding the depth camera's resolution or the number of sensors could reduce noise and improve object position estimation.

# 7. References

Ali, M. H., Aizat, K., Yerkhan, K., Zhandos, T., & Anuar, O. (2018). *Vision-based robot manipulator for industrial applications. Procedia computer science, 133, 205-212.* <https://doi.org/10.1016/j.procs.2018.07.025>

Haouala, Ines. (2021). Computer Vision Enabled Industrial Robot Manipulator for Sorting Operation. 10.13140/RG.2.2.10980.50562.

Tian, H., Song, K., Li, S., Ma, S., Xu, J., & Yan, Y. (2023). *Data-driven robotic visual grasping detection for unknown objects: A problem-oriented review. Expert Systems with Applications, 211, 118624.*

Zhou, X., Lan, X., Zhang, H., Tian, Z., Zhang, Y., & Zheng, N. (2018). \*Fully convolutional grasp detection network with oriented anchor box\*. arXiv. <https://doi.org/10.48550/arXiv.1803.02209>