# MINI ROBOT PROJECT
# ROS Entegration

## Table of Contents

## List of Figures

## List of Tables

# 1 Introduction

BRIEF INTRODUCTION

# 2 Getting Started

GETTING STARTED

## 2.1 Installation

To be able to get start the ROS part of project, followings must be installed.

> **Note**
>
> Be sure that you have a compatible version of Ubuntu for ROS2 Humble. If not, it should downloaded to Docker.This document doesn't include Docker installation instructions, but it can be found easily on the internet.

### 2.1.1 ROS2 Humble

Official installation can be followed: `https://docs.ros.org/en/humble/Installation/Ubuntu-Install-Debians.html`

Ad "opt/ros/humble/setup.bash" to the bashrc to source the enviroment for ROS when the restart, otherwise one must source in every case:

```
sudo nano ~/.bashrc
```

Listing 1: Source setup

### 2.1.2 Clone the Mini Robot ROS Repository

Clone the repository containing the mini robot's ROS packages:

```
git clone https://github.com/yourusername/minirobot.git
```

Listing 2: Cloning the Mini Robot Repository

### 2.1.3 Setup for Workspace

After downloading workspace you need to

```
cd MiniROS/dew_ws
colcon build --symlink-install
source install/setup.bash
```

Listing 3: Setting up Workspace

Note

- `colcon build` is used to build packages in a ROS 2 workspace. It compiles the source code, generates necessary build files, and prepares the packages for use. YOU SHOULD RUN EVERY TIME WHEN YOU MAKE CHANGE (With adding ”–symlink-install” no need to do when there is adjustment on pre existing file, run when there is new file,)
- `source` is used to set up the environment for a ROS 2 workspace. Running `source install/setup.bash` configures the shell to recognize the built packages, enabling you to run nodes and use the tools provided by the packages. YOU SHOULD RUN EVERY TIME YOU OPEN A NEW TERMINAL

### 2.1.4 Additional Setup

Some simulation enviroments and packages must be installed to run the project. I won't directly list them. You can follow the errors and install these when required.

If you have done all the installation until that point, lets run our first launch file:

```
ros2 launch mini_robot letsStart.launch.py
```

Listing 4: Running the Launch File

## 3 Project structure

The project is divided into four main sections, as shown in the figure below:

- **mini_robot**: This section contains configurations and description of the mini robot and the core functionalities.
- **mini_control**: This section includes the control logic for the mini robot. It manages the robot's movements and responses based on sensor inputs and tracking data.
- **aruco_tracker**: This section is responsible for detecting and tracking ArUco markers. It includes nodes and configurations for processing camera input and identifying marker positions.
- **ball_tracker**: This section handles the detection and tracking of a ball. It includes the necessary algorithms and nodes to track the ball's movement using image processing techniques.

The following sections will show the project structure and give small examples. We will start with using `mini_robot`, but if you are familiar with ROS and want to directly start using it with hardware, you can skip the `mini_control` section.

THE REST WILL BE UPLOADED SOON