Task 1 - Supply Bot

Task 1.3 - Bot Traversal

Introduction

The objective of this task is to trace and track a bot/ArUco marker moving around the circular Arena on its white concentric circular ring that lies placed between the purple and black concentric circular rings.

Before understanding task problem description, use all the additional resources provided in the **Task Description** folder, and for installing ArUco library make use of *"ArUco_library.pdf"*

Problem Description

Remember that all file and folder paths in your program should be relative. The video to work on is present in **"Videos"** folder. All your output files must be generated in **"Generated"** folder. Write your code in the placeholder file, *"main.py"* which can be found in **"Codes"** folder. All the resource related to ArUco are provided

The video present in the **"Videos"** folder is:

- *aruco_bot.mp4*

A single frame will look like the one shown below in Fig.1. and the video is nothing but a collection of all such images/frames over time:

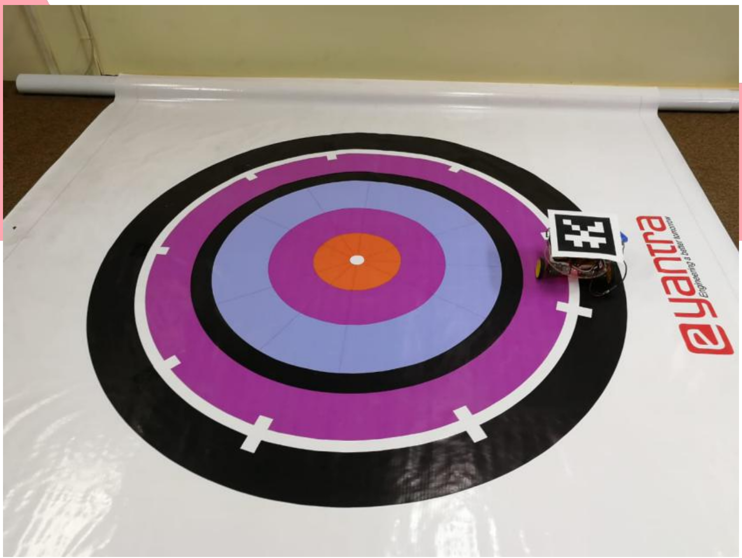


Fig.1.

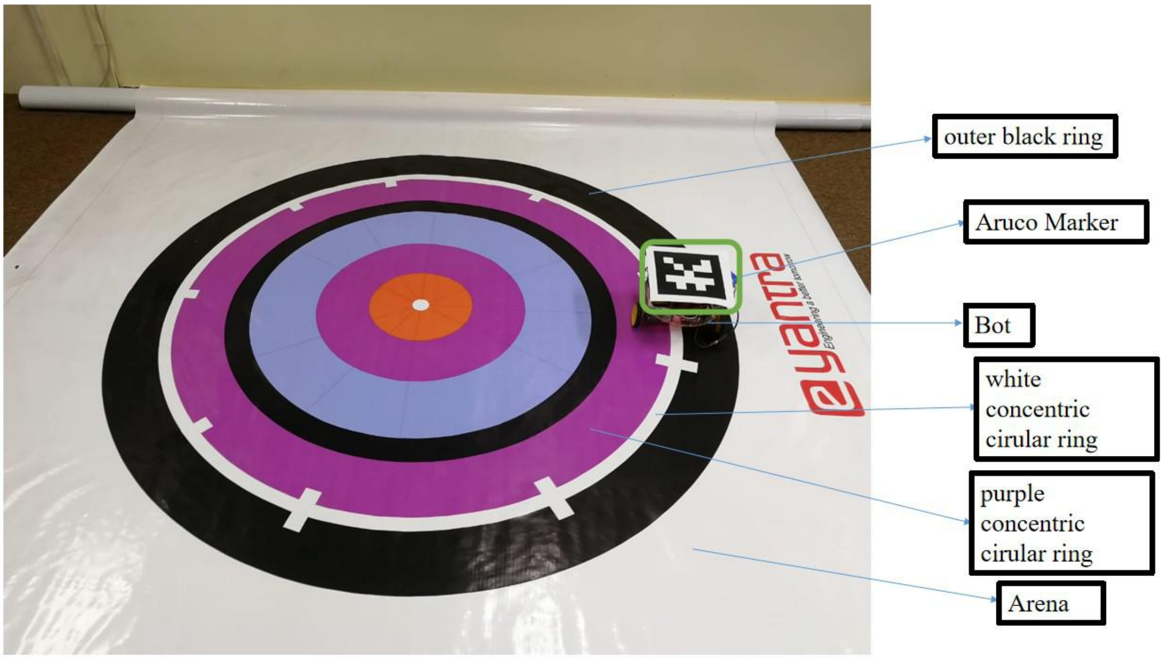


Fig.2.

Fig.2. shows all the corresponding components of a single frame as shown in Fig.1.

NOTE: All the components of the frames in the video are constant except the bot and the ArUco marker placed on it.

Programming Instructions

To complete the task, put your program logic in the *"main.py"* which can be found in **Codes** folder. The skeleton code in *"main.py"*, on running, takes in an input which is a particular frame in the video*("details can be found in comments of code")*, on this frame the team needs to do the following steps in order to complete the task.

For example, if we consider the 31st second frame, you will need to perform the following tasks:

1. Extract the frame at the start of the 31st second.
2. Adjust the contrast to make it as clear as the scene in Fig.1
3. Use filters to remove motion blur
4. Detect the id and angle of the ArUco on the bot
5. Write your results in ***"output.csv"***

NOTE: Remember we will execute the code during evaluation, during our run we will enter any second as input and evaluate your output at that frame.

On successful execution of *"main.py"*, the program will write into ***"output.csv"*** found in the **Generated** folder.

For example: if the ArUco ID detected is 70 at an angle of 167°

Image Name	ArUco ID	ArUco Angle
ArUco_bot.jpg	70	167

<u>NOTE</u>*:* Make sure the skeleton code is not modified, do all the coding in the specified section in *"main.py"*

Along with the ***"output.csv",*** also submit the processed image with ***ArUco ID*** and ***angle*** overlaid on it (Fig.4)**,** in the **Generated** folder of **Task1.3** with the name ***"aruco_with_id.png"***.

The ArUco provided to you is of dictionary **5x5_50** and the output overlaid with angle and id along with the sample of the same ArUco is provided in Videos folder.

Codes for detection, generation and associated library are provided in the Codes folder and are named **"aruco_detection.py", "aruco_generation.py"** and **"aruco_lib.py"** respectively. You are supposed to import and use these files in main as required in the process to solve the task.

Additional resources of ArUco and its concepts, links to documentation are given in extra documents of this folder (4 in all).

The ArUco on the bot is as shown in Fig.3. below:

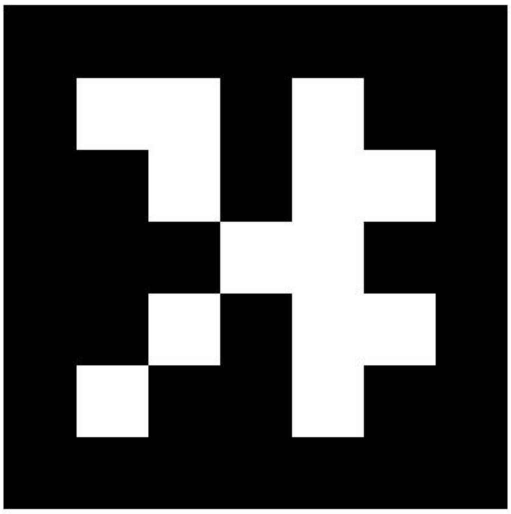
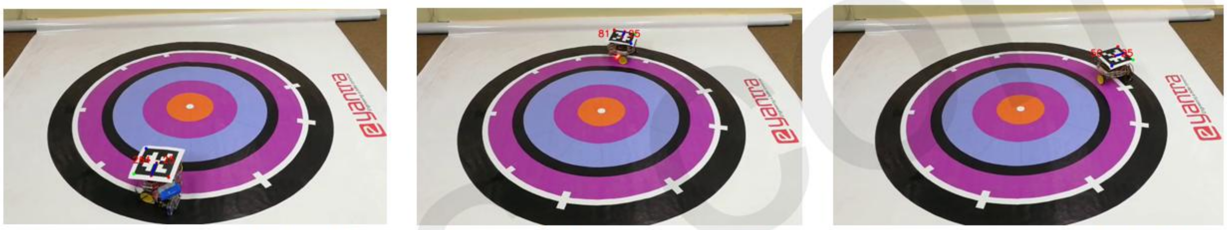


Fig.3.

The desired output with angle and ID overlaid is as shown in Fig.4. below:



Frame at 5th second - Angle is 264 | Frame at 42nd second - Angle is 81 | Frame at 31st second - Angle is 50

Fig 4: Sample outputs at 5, 31 and 42 second frame, after following the steps mentioned in "Programming Instructions"

Understanding *"main.py"*

In ***"main.py"*** we will find two important functions **def main(val)** and **def process(ip_image)** along with other declarations and initialization

def main(val) function has all the required statements to read video and read a particular frame which is decided by "val", the value will be passed at run time (when we run ***"main.py"***)

def process(ip_image) this is were the team needs to write their logic to complete the task

NOTE: Other then making changes to **def process(ip_image),** if is NOT REQUIRED to modify ***"main.py"***

Warnings

1. **IMPORTANT**: The code and documents you submit should be YOUR work in YOUR WORDS. To avoid any copyright violations, you must NOT copy phrases or code snippets directly from manuals or web.
2. The team should NOT mail or upload the document, code or folders anywhere else, except on the portal.
3. Teams failing to submit the document, code or folders by the deadline will lose the marks for this task.
4. e-Yantra WILL NOT entertain any request for extension of deadline for uploading the task.
5. e-Yantra holds complete discretion to disqualify a team if any foul play is suspected.