Task 1.1: Problem Statement

blogpost-style, resources, task-1-1

aayushi e-Yantra Staff Sep '21

Table of contents

Task 1

Welcome to Task 1 of Strawberry Stacker!!

The task is divided in two sub tasks and a bonus task

1. Task 1.1
2. Bonus Task
3. Task 1.2

Task 1.1

Overview

In task 1.1, we will be learning the following:

- Basic Image Processing techniques using OpenCV & Python
- Generating and detecting an ArUco marker
- Finding position and orientation of an ArUco marker.

Prerequisites

First things first, lets get the resources in place, install the necessary packages and go through the learning resources needed to complete the task

Installations

We assume that you have successfully installed and setup the software and packages as instructed in Task 0.

Note: Installation of px4 is not required in task 1, it will be required in further tasks and instructions will be provided in detail. Do not worry if the errors in px4 installations are not solved. If your px4 installations were not completed then create a separate workspace for this task and then follow the instructions. Otherwise you can use the previous catkin workspace.

You need to install the strawberry_stacker package in your *catkin_ws*

Open a new terminal window and type the following command

```
cd ~/catkin_ws/src
git clone https://github.com/erts-RnD/strawberry_stacker.git
```

```
cd ..
catkin build
```

Make sure your build succeeds without errors, if you face any errors resolve them before proceeding.

Learning Resources

Refer to the learning resources provided **here** before proceeding.

Additional tips

ArUco markers have a distinct id and the ArUco library (refer Tutorials) gives us the pixel position of the four corners and the center position of a marker. So this information helps you in finding the position and orientation of a marker. However, there must be a relative scale to measure the orientation of the marker. Orientation of the marker can be measured with respect to horizontal or vertical position. Let's fix the horizontal position as 0° and increasing upto angle 359° in anti-clockwise direction as shown in Figure 1:

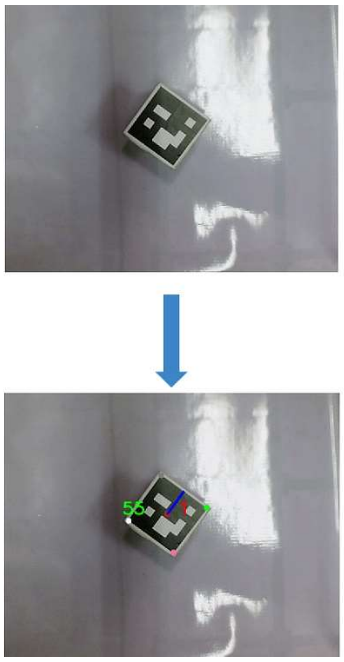

Figure 1: ArUco marker detection

Features in the resulting detected image as shown in Figure 2 are:

- Gray dot(125,125,125)(B,G,R) corresponds to top-left of the ArUco marker
- Green dot(0,255,0) corresponds to the top-right of the ArUco marker
- Pink dot(180,105,255) corresponds to the bottom-right of the ArUco marker
- White dot(255,255,255) corresponds to the bottom-left of the ArUco marker
- Red dot(0,0,255) corresponds to the center of the ArUco marker
- Blue line (255,0,0) corresponds to the line joining between the centre of ArUco marker and the mid point of the line joining top-left and top-right. This line gives us the indication of the orientation of the ArUco. The angle between this line with the horizontal axis gives us the required angle
- The upper thick black line indicates the horizontal line. Angle increases till 359° in anticlockwise direction. 55° is the angle between the blue line with this black line.
- Number in Red colour corresponds to the ArUco id
- Number in Green colour corresponds to the Angle of orientation of the ArUco marker
- Watch this **video** of ArUco detection for understanding the orientation scale better.

Problem Statement

Write Python script for Detecting ArUco markers . The Resulting image must have ArUco markers marked as shown in Figure1. Hence the resulting image will have ArUco markers with:

- Gray dot indicating top-left

- Green dot indicating top-right
- Pink dot indicating bottom-right
- White dot indicating bottom-left
- Red dot indicating center of the ArUco
- Blue line joining center of ArUco marker and the mid-point between top-left and top-right
- ArUco id number in RED colour
- Orientation of the ArUco in degrees in GREEN colour

Procedure

Step 1. Open *aryco_library.py* present in *scripts* folder.

Step 2. Complete the following functions:

- ***detect_ArUco(img)***
    - Functionality :
      → Detect id and corners of all the ArUco markers in the test image
    - Arguments :
      → img: the test image which needs to be tested
    - Return :
      → Detected_ArUco_markers: Dictionary in which each id (keys) corresponds to its corners(values).
- ***Calculate_orientation_in_degrees(Detected_ArUco_markers)***
    - Functionality :
      → Calculate the orientation of all the ArUco markers (degrees) in the test image relative to the scale as mentioned in Figure 1.
    - Arguments :
      → Detected_ArUco_markers: the Dictionary returned by detect_ArUco(img) function
    - Return :
      → ArUco_markers_angles: Dictionary in which keys are the ArUco ids and the values are the corresponding angle of ArUco's in degree
- ***Mark_ArUco(img,Detected_ArUco_markers,ArUco_markers_angles)***
    - Functionality :
      → Mark ArUco markers in the test image as per the instructions given in Figure1.
    - Arguments :
      → img: Test image
      → Detected_ArUco_markers: Dictionary returned by detect_ArUco(img) function
      → ArUco_markers_angles: Dictionary returned by Calculate_orientation_in_degress(Detected_ArUco_markers)
    - Return :
      → img: the resultant image after marking the ArUco's.

Step 3. After completing *aruco_library.py*, run the Python script *aruco_detection.py* which is present in the same folder as that of *aruco_library.py*. Don't change anything in *aruco_detection.py*. Once you successfully run *aruco_detection.py* you will find two files namely Result_image_1.png and Result_image_2.png in the folder scripts in your ROS package.

Step 4. Open the result images to ensure that all the attributes that mentioned in Figure1 is marked on all the ArUco markers in the test images.

Expected Output

Skip to main content

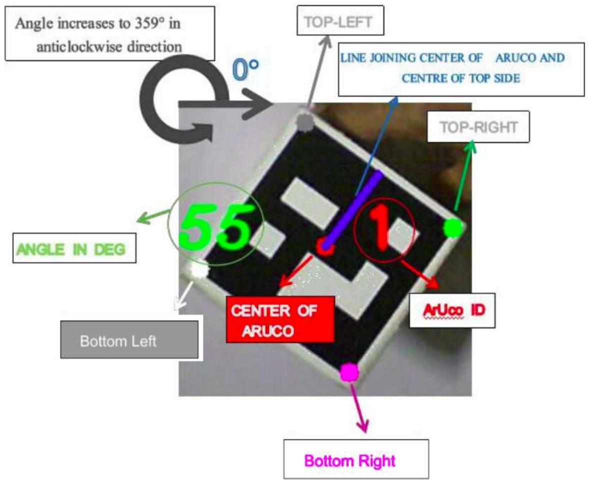
Angle increases to 359° in anticlockwise direction
0°
TOP-LEFT
LINE JOINING CENTER OF ARUCO AND CENTRE OF TOP SIDE
TOP-RIGHT
ANGLE IN DEG
55
1
Bottom Left
CENTER OF ARUCO
ArUco ID
Bottom Right

Figure 2: Expected Output

Submission Instructions

- **Copy aruco_library.py, Result_image_1.png and Result_image_2.png to a folder named <team_id>Task1_1.**
- **Instructions for uploading the task will be provided on the portal.**

Bonus Task

Write a Python script for detecting ArUco markers from a video stream. You can get the video from **here** . You can take help from the aruco_detection.py script.

Expected output

Your output should look like the video below:

Note

This topic is closed for responses for a better workflow, please ask your doubts in the **QnA category** using the tag `task-1-1`. Thanks and all the best.

URGENT! Error in task 1.1 problem statement
No resources for Task 1
Task 1 : Getting Started with ArUco and ROS

Skip to main content

CLOSED ON OCT 18, '21

UNLISTED ON OCT 23, '21