Orangewood.

Confidentiality Note

This test and all its contents are the property of Orangewood Labs. This is a confidential document intended for evaluation of a candidate for a suitable role at Orangewood. It must not be shared with anyone other than the candidate. The candidate must not release it in the public domain or share it with any other person. This document must be destroyed by the candidate after its intended purpose has been served.

In the submission, please acknowledge any third-party resources that were used; e.g. links to websites, videos, code, etc.

Prerequisites

To complete these you would require a system running Linux (Recomended Ubuntu) and have ROS1 installed. Additionally libraries like OpenCV will also be required. Please follow the instructions below to get started -

- ROS1 Installation
- Catkin Installation
- Turtlesim Installation
- OpenCV Installation

Problem Statement

TASK

OVERVIEW

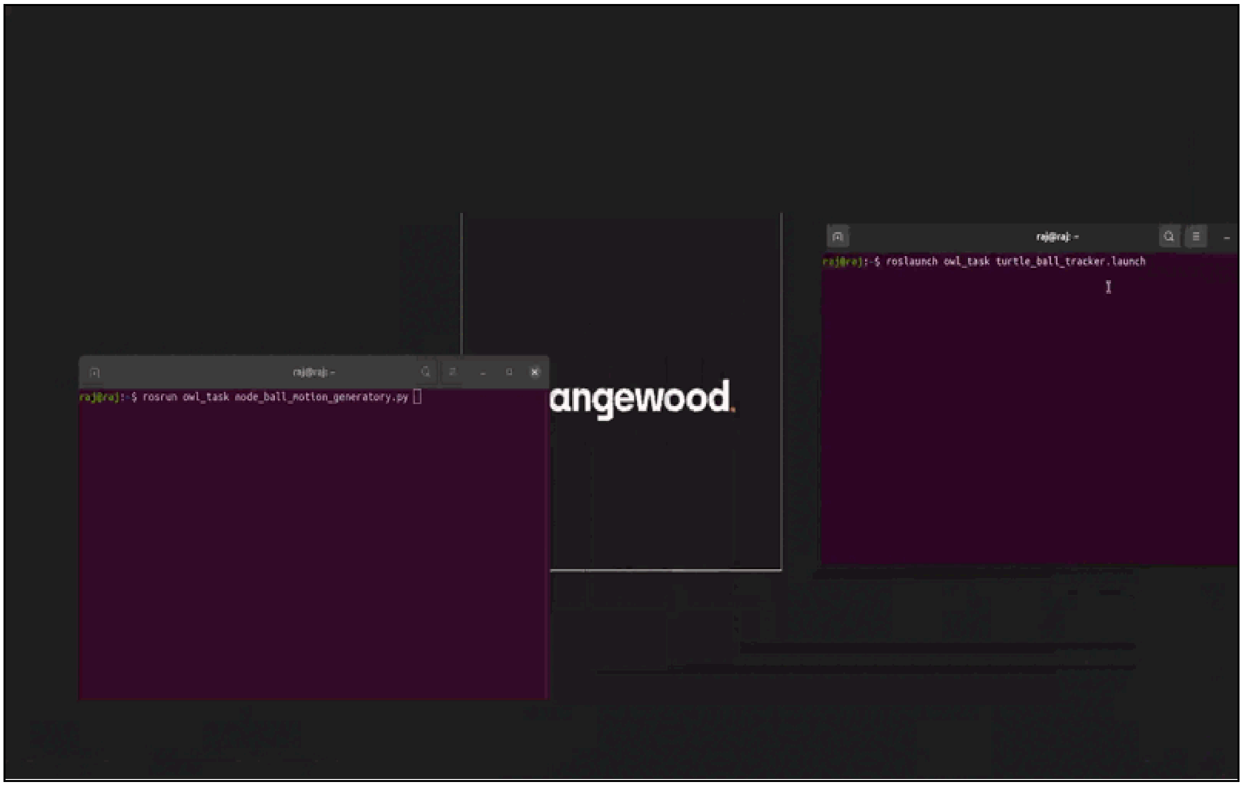
Computer Vision is an essential part of robotics as it helps the robot extract relevant information from camera data to understand its environment. Applications range from inspecting manufactured parts for defects to detecting pedestrians in autonomous vehicles. The objective of this task is to use computer vision to track a green ball that is continuously traveling in a lemniscate curve and write a control algorithm that makes the turtle inside turtlesim follow the ball's movements in realtime. You will have to write three custom ROS nodes that communicate in tandem to achieve this. Detailed description provided below -

- Task 2a : Motion Generation
  - Using OpenCV, generate a video feed of a green ball continuously tracing a lemniscate curve **(frame size: 500 x 500px).** Consider the centre of the frame to be the lemniscate origin, and the foci to be 150px.
  - Publish this video feed on a ROS Image topic with any arbitrary name like **/image/ball_animation.**
  - View the output in RViz or rqt_image_viewer

- Task 2b: Ball Tracking
  - Create a node that subscribes to the generated video feed and uses OpenCV to continuously compute the (x,y) centre coordinates of the ball then publish it to the topic '**/turtle/move pose**' using the PoseStamped msg type. Make sure to correctly map the x,y coordinates from the image to turtlesim coordinates.

- Task 2c: Turtle Control
  - Create a control node which subscribes to the topic '**/turtle/move pose**' and makes the turtle in turtlesim follow the waypoints provided by the ball tracking node. Thus the turtle's motion should also follow a lemniscate curve. Do note that while you are providing poses, turtlesim only accepts cartesian velocity (Twist) commands. It is up to you on how you implement this.

Expected Outputs



```
raj@raj:~$ rosrun owl_task node_ball_motion_generatory.py
```

```
raj@raj:~$ roslaunch owl_task turtle_ball_tracker.launch
```

Submission Instructions

Compile all developments inside a single folder, accompanied by a screen recording. Include a README.MD File for the purpose of installation and usage of your code.Package everything in a zip file.

You can further improve your chances of selection,

- By ensuring your code is well-documented and follows best practices.
- By completing both tasks in C++ instead of Python. But do note that correctly completing the task is far more important than the choice of language.