EE346 - Mobile Robot Navigation and Control (Draft)

Winter 2022
Laboratory #3 (5%)
Due Date: TBA
Camera Calibration and AR Tag Recognition

Objectives

- Learn how to calibrate a camera
- Study visual object detection of AR tags
- Perform pose estimation with the help of AR tags

Procedure

In this lab, you will first learn to how to calibrate your TurtleBot3 camera using the ROS package camera_calibration. Then you will proceed to study how to detect AR tags and estimate camera pose with respect to an AR tag using the ROS wrapper for Alvar, ar_track_alvar. Finally, you will integrate the AR tag detection and pose estimation functions with the line following function in Lab 2 so that your robot can detect AR tags around the track, signal the detections, and stop in front of a specific designated AR tag.

Part I: Camera calibration

In this part of the lab, you will calibrate the monocular camera on your Turtlebot3. Use the check board pattern that is provided to you, and follow the instructions at http://wiki.ros.org/camera_calibration for running the camera calibration node. Make sure that your camera node is properly publishing images to the topic /camera/image_raw. Study the tutorial at http://wiki.ros.org/camera_calibration/Tutorials/MonocularCalibration, to perform the actual camera calibration. Complete the steps in the tutorial, and save the calibration results in a yaml file (typically named camera.yaml) for use in the next two parts. Inspect camera.yaml to understand its contents, which define your camera's characteristics: image resolution, calibration matrix, distortion coefficients, rectification matrix, and projection matrix.
ROS camera calibration package is based on OpenCV, and this page explains further details on if you are interested: https://docs.opencv.org/master/d4/d94/tutorial_camera_calibration.html.
Also you can read this tutorial to understand the camera calibration algorithm and process: https://learnopencv.com/camera-calibration-using-opencv/.

Upon completion, show the calibration parameters in the saved yaml file to a TA, and explain what the numbers mean.

Part II: AR tag detection and pose estimation

In this part of the lab, you will study how to recognize and perform pose estimation of AR tags, a fiducial (reference) marker system to support augmented reality (AR). We will use the popular ROS package, ar_track_alvar, a "wrapper" around the AR tag library called Alvar, in our experiments. First study the Wiki page: http://wiki.ros.org/ar_track_alvar, to install the package. You can either print your own AR tags for numbers from 0 to 9 (10 different numbers) or use the ones we provide to you. With the camera node running, run the node ar_track_alvar and display/visualize the two topics published by the ar_track_alvar node: the first on the topic of visualization_marker indicates which AR tag (of the 18) is being recognized, and the second on the topic of ar_pose_marker shows the pose of the camera with respect to the AR tag in terms of a tf message. Make sure that you are able to recognize the AR tags correctly and verify the pose in the tf message. Note that the Wiki page above is for a different robot called PR. To modify the launch file for the ar_track_alvar node to work for your Turtlebot3, refer to the blog at http://ros-robotics.blogspot.com/2015/04/recognize-ar-tags-with-ros.html for additional information.

Upon completion, show the operation of the AR code detection by capturing images by the camera on the robot, placing one of the AR tags in front of the camera, and printing on the screen both the AR tag number and the relative pose from the camera to the AR tag. Make sure you can explain the Pose message that describes the pose.

Part III:

TBA

Marking