RBE 450X – Homework 3

In this homework, we will apply various computer vision techniques to detect several image features on an object. These image features are not only important to understand the object geometry (so that this information can be used for planning and executing robotic manipulation strategies), but also would allow us to implement vision-based control algorithm that we will cover in the upcoming weeks. We have started implementing a part of this assignment in the lab sessions, so you can continue directly from where you were left off.

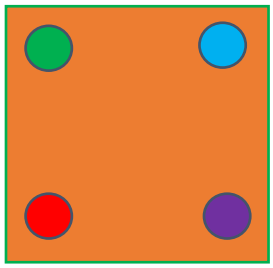
This HW needs to be implemented in ROS, Gazebo and OpenCV. If you are using python please be sure to install opencv using "sudo apt-get install python3-opencv"

Please carefully read the deliverables of each step. Please submit one single PDF file with the screenshots of the required images.

Step 1 (1.5 pts): Download the gazebo robot package from the week 4 module in Canvas (be careful, not the ones in the week 3 module). Follow the instructions and spawn your robot in gazebo.

Deliverable: Take a screenshot of the spawned robot in the gazebo environment.

Step 2 (4 pts): From Canvas, download the URDF file of the plate that has four colors as below:



Also download the .py file that will be used to spawn this plate to the Gazebo environment.

Using these two files, spawn the plate to the Gazebo environment. You can check this tutorial for some tips.
https://automaticaddison.com/how-to-load-a-urdf-file-into-gazebo-ros-2/

Move the robot in such a way that the plate is completely visible from the camera view and robot is NOT in a singular position (not completely straight). You can send position commands using the "ros2 topic pub…" command. Please check the ROS tutorials for more information.

View the images that is sent by the simulated camera using the image_view package of ROS. If you do not have this package, you need to install it using “sudo apt-get install ros-humble-image-view”. You can run image view via terminal. Do not forget to remap the topic of the package with the argument “-r image:=whatever_the_name_of_the_image_topic_is”.

Deliverable: Take a screenshot of the object and robot in the gazebo environment AND the screenshot of the image_view output.

Step 3 (1.5 pts): Download the corresponding opencv test package. (There is one test package for c++ users (opencv_test) and one test package for python users (opencv_test_py); choose the one according to the programming language of your choice). Examine the code. These packages get the camera images from the simulator and publish them back to ROS. Run one of the example codes, display their output via the image_view package (caution you should remap the “image” topic to “output_image” while using image_view).

Deliverable: Take the screenshot of the image_view window and add it to your report.

Step 4 (2 pts): Now modify these test files so that you apply color thresholding to the image using OpenCV functions (please see an example here: https://ckyrkou.medium.com/color-thresholding-in-opencv-91049607b06d. For the rest of the HW, I will not be providing examples, but will expect you to find them yourselves.)
Deliverable: Find the center of each circle by averaging all the pixels that belong to that color. Write down the numbers AND include a screenshot to your code in your report

Step 5 (1 pts): Apply canny edge detection algorithm to the captured image using OpenCV functions.
Deliverable: Take the screen shot of the resulting edge image and include it to your report AND include a screenshot to your code in your report

BONUS: Step 6 (2 pts): Apply Harris corner detection algorithm to the captured image using OpenCV functions.
Deliverable: Mark the corners on the image, take the screen shot of the resulting edge image and include it to your report AND include a screenshot to your code in your report

BONUS: Step 7 (2pts): Apply Hough circles algorithm to detect the circles and find their centers.
Deliverable: Write down the centers of the detected circles AND mark the detected circles, take the screen shot of the resulting edge image and include it to your report.