PREFACE

Preparing laboratory experiments can be time-consuming. Quanser understands time constraints of teaching and research professors. That's why Quanser's control laboratory solutions come with proven practical exercises. The courseware is designed to save you time, give students a solid understanding of various control concepts and provide maximum value for your investment.

Quanser QBot 2 for QUARC courseware materials are supplied in a format of the Instructor and Student Manuals.

This courseware sample is prepared for users of The MathWorks's **MATLAB®/Simulink®** software in conjunction with Quanser's **QUARC®** real-time control software.

The following material provides an abbreviated example of lab experiments for the QBot 2 for QUARC experiment. Please note that the examples are not complete as they are intended to give you a brief overview of the structure and content of the course materials you will receive with the plant.

TABLE OF CONTENTS

1. Introduction to Quanser QBot 2 for QUARC Courseware Sample

Quanser courseware provides step-by-step pedagogy for a wide range of control challenges. Starting with the basic principles, students can progress to more advanced applications and cultivate a deep understanding of control theories. Quanser QBot 2 courseware covers topics, such as:

- differential drive kinematics
- forward and inverse kinematics
- dead reckoning and odometric localization
- path planning and obstacle avoidance
- 2D mapping and occupancy grid map
- image acquisition, processing and reasoning
- simultaneous localization and mapping (SLAM)
- vision guided vehicle control
- high-level control architecture of mobile robots

2. Instructor Workbook Table of Contents

The full Table of Contents of the Quanser QBot 2 for QUARC courseware is shown here:

Experiment 4 – Path Planning

PATH PLANNING

3. BACKGROUND SECTION - SAMPLE

Differential Drive Kinematics

The QBot 2 is driven by a set of two coaxial wheels. These wheels are actuated using high-performance DC motors with encoders and drop sensors. To determine the relationship between the independent motion of the two wheels and the motion of the overall robot, we begin by modeling the motion of the robot about a common point. Let the radius of the wheels be denoted by r, and the wheel rotational speed be denoted by ω_L and ω_R for the left and right wheel respectively. The linear speed of the two wheels along the ground is then given by the following equations:

$$v_L = \omega_L r \quad (1.1)$$

$$v_R = \omega_R r \quad (1.2)$$

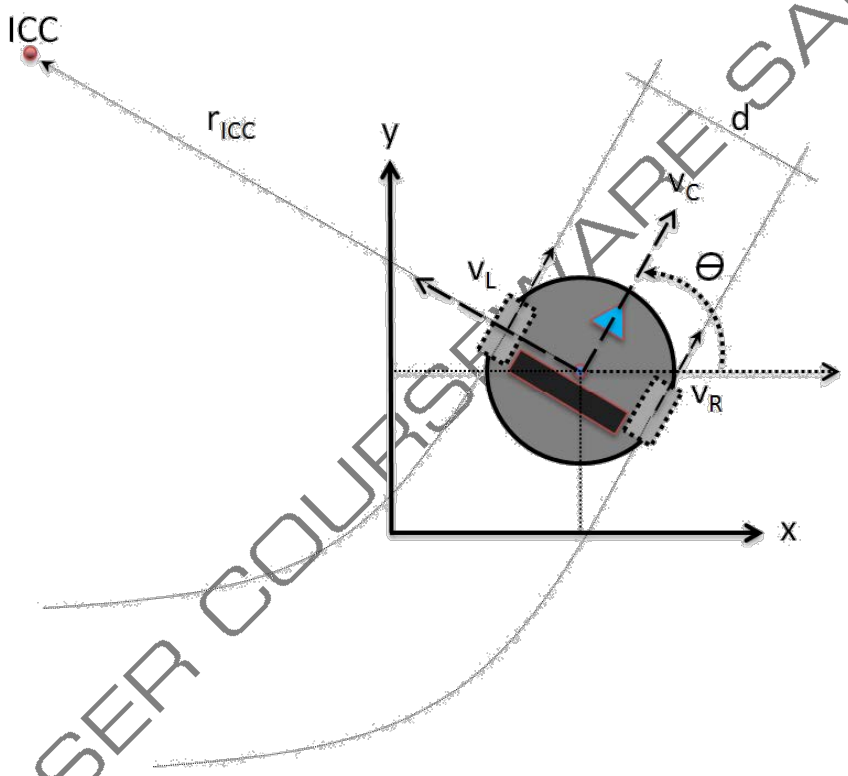

Figure 1.1: Quanser QBot 2 Mobile Platform Reference Frame Definitions

Assuming there is no wheel slippage, the QBot 2 can move along the horizontal plane in straight or curved trajectory, as well as spin on a spot, by varying the relative speed between the left and right wheels.

Since we are assuming that the wheels are not subject to slip, the motion of the wheels are constrained to move along their forward and backward directions. This, together with the inherent constraint that is imposed by the robot chassis coupling the two wheels together, means that all robot chassis rotations must be about a point that lies along the common wheel axis. For example, if only one of the two wheels rotates, the robot would rotate (pivot) about the non-moving wheel. On the other hand, if both wheels rotate at the same speed, the robot rotates about a point infinitely far from the robot. This center of rotation is known as the *Instantaneous Center of Curvature* (ICC).

Kinematic Model

Let r_{ICC} be the distance measured from the center of the robot chassis, which is halfway between the left and right wheels, to the ICC. If d is the distance between the left and right wheels, θ is the heading angle of the robot, and v_C is the (forward/backward) speed of the robot chassis center, the motion of the QBot 2 chassis can be summarized in the following equations:

$$v_C = \dot{\theta} r_{ICC} \tag{1.3}$$

$$v_L = \dot{\theta}\left(r_{ICC} - \frac{d}{2}\right) \tag{1.4}$$

$$v_R = \dot{\theta}\left(r_{ICC} - \frac{d}{2}\right) \tag{1.5}$$

Notice that v_C, v_L and v_R are all defined along the same axis, which lies in the forward/backward direction of the chassis. Given the wheel speed, v_L and v_R, the robot speed, v_C, the angular rate, $\omega_C = \dot{\theta}$, and the distance from ICC, r_{ICC}, we can relate the motion of the wheels to the motion of the robot using the following kinematic model for the differential drive system:

$$v_C = \frac{v_R + v_L}{2} \tag{1.6}$$

$$\omega_C = \dot{\theta} = \frac{v_R - v_L}{d} \tag{1.7}$$

$$r_{ICC} = \frac{d}{2}\frac{(v_R + v_L)}{(v_R - v_L)} \tag{1.8}$$

4. In-Lab Exercise Section - Sample

Image Processing – Thresholding

The first controller model that is used for this lab is "QBot2_Image_Tresholding.mdl" shown in Figure 2.3

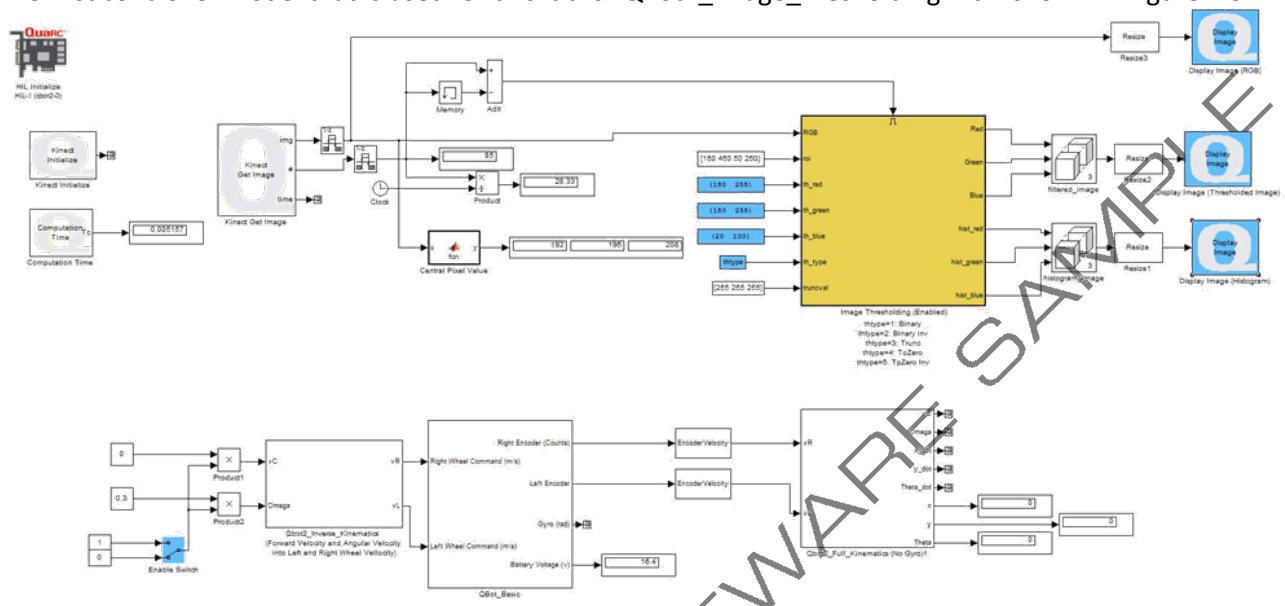

Figure 2.1: Snapshot of the controller model "QBot2_Image_Tresholding.mdl"

The *Image Thresholding* block, in yellow, implements the thresholding algorithm using the different methods described in the Background section. This block receives the frame, region of interest (roi), thresholds on all three color channels (th_red, th_green, th_blue), the method (th_type) and the truncation values (thuncval) required for the truncation approach. This block outputs the processed image through RGB channels as well as the histograms.

Follow the procedure outlined below. Make observations about the effect of the algorithms on the generated images, and answer the associated questions.

1. Open the supplied model, QBot2_Image_Thresholding, and make sure the manual switch (Enable switch) is
2. set to zero. Compile the model and run it.
3. Double click on the three *Display Image blocks* in your model.
4. Find or create a coloured sheet of paper, and cut out a 10cm x 10cm square.
5. Bring the colored sheet close to the QBot 2, and use the RGB image to estimate the RGB values of the paper. To do so, change the mouse function to "Data Cursor" as shown in Figure 2.2. Select a point and move it to different locations on the RGB image, and note the RGB values for a few points on the card. Tilt the card in various orientations and observe the changes in the measured colours. Then define a range for the RGB values that represent the colour of the card.
6. Inspect the "Display Image (Histogram)" window, and observe the changes in the output when you bring objects with different colors, including the colored piece you just made, in front of the robot.

Can the histogram help you find the [min max] ranges (the histogram is scaled by a factor of 10)? Save a snap-shot of the image in each case, and discuss the results.

7. Using the range that you found, set the [min max] values of the three threshold parameters, th_red, th_green and th_blue.
8. If the filtering does not seem good enough, try tuning the threshold values until the coloured piece of paper is completely filtered out.
9. Double-click on the "th_type'' variable, and select the various options one-by-one, observing what each method does to the results. Save a snap-shot in each case, and discuss the results.

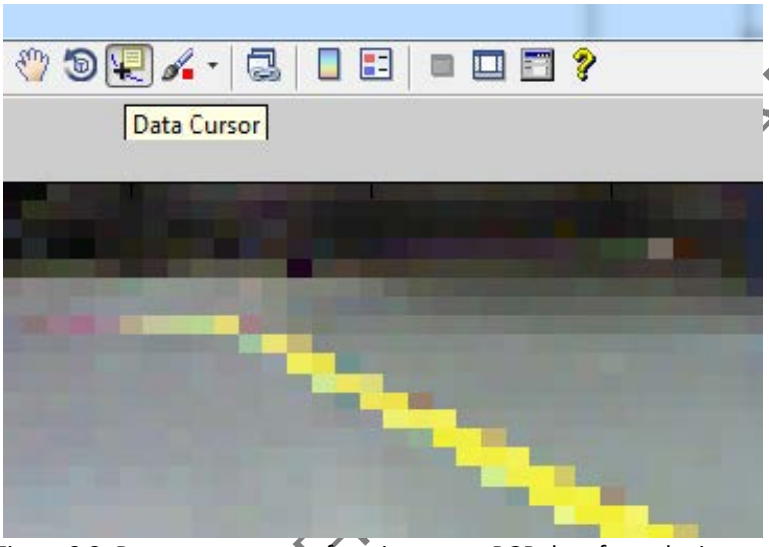

Figure 2.2: Data cursor mouse function to get RGB data from the image