MEM380/800: Applied Autonomous Robots II

Assignment 5: Getting to Know the SRV-1

Due: 2/11 at 11:59pm EST

1 Objective

The objective of this lab is to familiarize you with the SRV-1s and to help you calibrate and derive a motion model for the robot. MATLAB control scripts for controlling the robot can be downloaded from the course website by following the Resources link to SRV-1 Toolbox. Information on how to use these MATLAB scripts are also provided. Remember, you are allowed to work in groups of 2.

2 Task 1: Getting to Know the SRV-1

2.1 Establishing a Connection with the Robot

All our robots are configured with *IP addresses* of the form

$$192.168.1.xx. \tag{1}$$

The IP address is the wireless network address of the robot. You will need to know this if you want to communicate with the robot. Pick a robot and note its ID number. To determine your robot's IP address, simply replace the xx in (1) with the robot's ID number. For good measure and easy reference, note the address below.

IP Address:

1. Remove the robot from power and turn it on. An indicator light should glow showing that the power is on. YOU SHOULD NEVER TURN ON A ROBOT WHILE IT IS CHARGING!
2. Open up the Command Prompt in Windows. It can be accessed through: Start Menu ⇒ Programs ⇒ Accessories ⇒ Command Prompt.
3. With the Command Prompt open, type the following into the black screen:

   ```
   ping <your robot's IP address here>
   ```

   Then press ENTER. At this point, you should receive some feedback in the Command Prompt: `Pinging <the IP address you entered> with 32 bytes of data:`
4. If all goes well, you should receive a response for all four ping attempts. This means that your robot is set to communicate wirelessly.
5. If for some reason the ping attempts do not go through smoothly, wait about 2 minutes and try again. Often it takes a minute or two for the robot to initialize once powered on. If this doesn't work check that you have the correct IP address, turn the robot off and then on again, *i.e.* recycle the power, and wait another minute before pinging.

6. You should be able to move the robot around the room and `ping` it successfully. Try this in several locations, including ones that seem obstructed. This will help you get an idea of what kinds of obstacles the wireless system can deal with. Ideally locations are where all 4 of the 32-byte packets go through.

7. After a successful ping attempt, write down the response time for each of the packets just so I have an idea of what kind of lag time your group dealt with:

2.2 The SRV-1 Console

This is a basic user interface provided by Surveyor which allows you to control the robot via the network from a desktop. While the SRV-1 Console is not a particularly useful control interface, it is a nice tool that can be used to verify that the hardware is working properly. Now that your robot is set to communicate wirelessly over the network, we are ready to try out this basic user interface.

Recommendation: Keep the Command Prompt window open. This will enable you to `ping` the robot and check your connectivity with it at all times. If you think your robot is not responding, you should always `ping` it first and make sure you are still connected to it before trying anything else. If the time delays get to be too long, this may pose a problem. In such situations, you may need to find a better location to work with the robot.

2.3 Setting Up the SRV-1 Console

1. Download `SRV1Console.zip` from the course website. Unzipped the files and into a folder called `SRV1Console`. Within this folder, look for a file entitled `srv.config` and open it with either WordPad or Notepad. The second to last line of this file should contain the following line:

   `network.srv.host=169.254.0.10`.

   Note: It is okay if the IP address doesn't match what is shown above exactly. Replace the listed IP with your robot's IP.

2. Open the SRV-1 Console by double-clicking the file `SRV1Console.exe` in the same `SRV1Console` folder. A window like the one shown in Figure 2 should pop up. In the drop–down menu, make sure you select the 'Network' option and click Connect. Once you have clicked Connect and image should show up in the center of the Console. The image is sent from the robot's camera. The frame rate is not going to be particularly high, but you can decrease the resolution to get a faster refresh rate.

3. Once the connection has been established between the Console and the robot, you are now able to use the buttons to command the robot to drive around, go fast or slow, etc. Try it out and see if you can get a sense of what each button does. Some of these do not work properly, but note the ones that do.

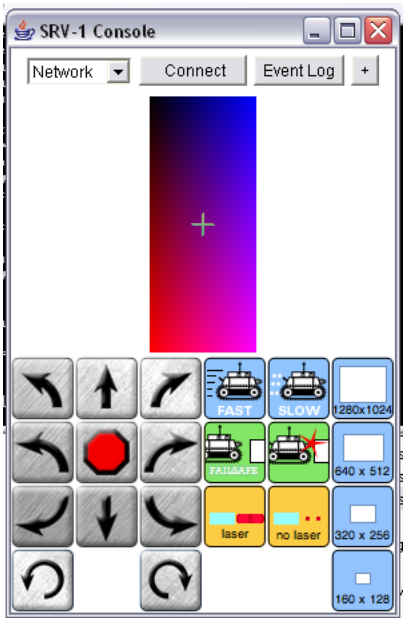

Figure 1: SRV-1 Console Application.

4. If at any time the feed times out, close the application, reset (turn off and on) the robot, and attempt to ping it again. When the pinging is successful, reopen the console and it should work again. Also, make sure that the wireless network is working properly and if it isn't, let one of the TAs know.

Play with the buttons and see what they do.

2.4 Controlling the Robot via MATLAB

The SRV-1 Robots have been configured with code written primarily in Java, but all of the programming and functions you will use are provided to you via MATLAB scripts. In this part of the lab, you will test the communication and your robot's execution of commands sent to your robot in MATLAB through the wireless network.

Recommendation: If you are having connection issues with the robot, you may want to consider pinging the robot frequently. An easy to do this is to use the `-t` tag, *i.e.* type `ping -t <robot ip>`. This will generate a continuous string of pings. While this may fix your network timeout issues, it will also drain your battery much faster, so make sure you use this selectively.

1. Open `basic_drive.m` in MATLAB . Make sure you read the comments carefully. Follow the steps outlined in the comments and fill in the section marked `%YOUR CODE HERE`. To execute simply run the script.

2. Your robot should complete the series of commands requested.

3. When you are sure that the robot has completed the movements, demonstrate this to a TA. You will not get credit for this portion unless the TA has seen your demo.

4. Finally open `figure8_drive.m` in MATLAB . Once again, make sure you read the comments carefully. This time you will be writing a program to drive the robot in a Figure 8 pattern.

5. Once again, when you are sure that the robot has completed the movements, demonstrate this to a TA. You will not get credit for this portion unless the TA has seen your demo.

Include in your Report: Include a brief description (¡ 1 paragraph) of how you determined the commands needed to drive the robot in a Figure 8 pattern. Additionally, include your code. Make sure every line of the code is clearly commented.

3 Task 2: Calibrating the Robot

To drive the SRV-1, we send the robot commands of the form m_R, m_L, and t where $-100 \leq m_R, m_L \geq 100$ and t denotes the duration time in seconds. (Note: In theory, if you set $t = 0$ this will result in the robot driving on forever. However, in practice, this simply drives the robot for a fixed amount of time whose value is set by the manufacturer. I suggest you always give the command with $t \geq 0$.) Recall, for a differential drive robot, the relationship between the forward velocity, v_f, and the angular velocity, $\dot{\theta}$ to the wheel speeds, $\dot{\phi}_R$ and $\dot{\phi}_L$, are given by

$$\begin{aligned} v_f &= \frac{r}{2}(\dot{\phi}_R + \dot{\phi}_L) \\ \dot{\theta} &= \frac{r}{l}(\dot{\phi}_R - \dot{\phi}_L) \end{aligned}$$

where r denotes the radius of the wheels and l denotes the axle length.

While we do not have access to $\dot{\phi}_R$ and $\dot{\phi}_L$ for our robots, we can interpret m_R and m_L as a proxy for $\dot{\phi}_R$ and $\dot{\phi}_L$. As such, in our calibration procedure, we would like to determine the following relationships:

$$\begin{aligned} v_f &= h_1(m_R + m_L), \\ \dot{\theta} &= h_2(m_R - m_L), \end{aligned}$$

where h_1 and h_2 denote functions that will be approximated from experimental data. The following sections provide steps that will help you determine h_1 and h_2 and thus a basic motion model for the robot.

For this exercise, you will need the following: a straight line marked with masking tape that is 1 meter in length. You will need a stop watch and some additional masking tape (this should be located in the same cabinets where the robots are stored). Start up MATLAB and get yourself set up so that you can command the robot via MATLAB . Once again, if you need help, go to the course website and look over the instructions provided.

3.1 Straight Line Motion

To determine h_1, you will need to select a set of values for m_R and m_L such that $m_R = m_L$ and m_R is between 0 and 100. You should select at least 5 pairs of values. Now command your robot to drive down the straight line that the TAs have marked out on the floor. Make sure you set your drive time long enough for the robot to be able to traverse a little farther than the full length of the line. Use your stop watch to time the amount of time the robot takes to drive the full length of the line, t_f. The forward speed of the robot is then given by $v_f = 1(m)/t_f(sec)$. Ideally, you would repeat this multiple times for every pair of values you have selected and record the mean and standard deviation of v_f for each m_R and m_L pair. However, for this assignment, you will only need to do it once.

Note: Those of you who feel comfortable with MATLAB should try writing your own MATLAB script for this task. However, I do provide a basic MATLAB program that will let you drive the robot down the straight line for the set of selected values. See `straightLineCalib.m` in `roboCalibMatlab.zip`.

Once you have collected this data, plot your results, *i.e.* generate a plot of v_f vs. average $(m_R + m_L)$. Can you discern a relationship between average v_f and $(m_R + m_L)$? Do you see a

pattern emerge? If the answer is no, you may need additional data points. If you think that is the case, select additional values for m_R and determine v_f following the procedure outlined above. Add the new data points to your plot. Do you think the relationship between average v_f and (m_R+m_L) is linear, quadratic or logarithmic? Do you see any saturation?

Enter your data into Excel and fit a curve to your data. To achieve this first plot your data using an `X-Y Scatter` plot. Once you have plotted your data, select `Add Trendline` from the `Chart` menu and select the type of curve to fit your data to. Before you select `OK`, click on the `Options` tab and make sure you check the boxes for `Display equation on chart` and `Disply R-squared value on chart`. The R-squared value gives us a measure of how well our data fits the particular trendline. The closer the value of R-squared is to 1, the better the fit.

3.2 Rotational Motion

Similarly, when calibrating for $\dot{\theta}$, select a set of values for m_R between -100 and 100 and set $m_L = -m_R$. Now command your robot to turn for a fixed time t_θ such that t_θ is long enough for the robot to turn a little over $360°n$, where n is an integer. Using a stopwatch, measure the amount of time it takes the robot to make exactly n full turns. Once again, you should select at least 5 pairs of values for m_R and m_L. Ideally, you would repeat the procedure multiple times for every pair of values and record the mean and standard deviation of $\dot{theta}$ for each m_R and m_L pair. However, for this assignment, you will only need to do it once.

Note: Since this task is very similar to the previous one, modify `straightLineCalib.m` to help you collect the necessary data.

Once you have collected this data, plot your results, and see if you can discern a relationship between average $\dot{\theta}$ and $(m_R - m_L)$? Remember, if you do not see a pattern emerge you may need more data points. Do you think the relationship between average $\dot{\theta}$ and $(m_R + m_L)$ is linear, quadratic or logarithmic? Do you see any saturation? Again, use Excel to fit a curve to your data.

For the report: Summarize all data you collected. List all m_R and m_L values you considered and the corresponding v_f and $\dot{\theta}$ you measured. Include the mean and standard deviation of v_f and $\dot{\theta}$ for each m_R and m_L pair. Include the curve fitting results you obtained from Excel. Discuss the fits you obtained. Lastly, make sure you include the following talking points in your discussion:

- How many data points did you chose and why? Do you think you obtained useful statistical results about the robot's motion behavior/model?
- Analyze the velocity graphs for rotational and straight-line motion. Does the behavior seem to follow the theory? Are the motor commands linear, piece-wise, or non-linear?
- What were the problems you had, if any, during this lab? How did you deal with them?

4 Extra Credit

Recall, for a differential robot, we can relate the wheel speeds to the robot's instantaneous radius of curvature as follows:

$$R_{ICC} = \frac{l\left(\dot{\phi}_R + \dot{\phi}_L\right)}{2\left(\dot{\phi}_R - \dot{\phi}_L\right)}.$$

Design a procedure that will enable you to obtain the necessary data to relate m_R and m_L to R_{ICC}. Modify `straightLineCalib.m` to enable you to collect this data. Include your results and a brief discussion.

For your report: Once again, clearly outline the assumptions and reasoning you used to arrive at your methodology. Present your results and make sure you provide the necessary data to back up your results and conclusions.

Final Words on the Report: Each team will only have to submit 1 report. This is NOT a lab or project report so you can keep your answers short but concise. Make sure you include your figures and brief explanations to all of them.