

CS4630 Robotics and Sensing

Project 5

Out: Mon., March 25, 2002

Due: Mon., April 8, 2002

1 Introduction

Sensor and actuator specifications are often provided by the robot manufacturer. However, in some cases these specifications are not provided or the programmer has some reason to believe that they are incorrect. Finding out the correct specifications is very important when faced with a new robot platform. This project requires you to find these specifications for the Descartes robots.

There are three problems in this project. The first two require you to estimate the forward movement and turn accuracies of the robot. The third problem uses the results of the previous two to achieve precise robot movements for an interesting application.

Before you jump to the problems, however, read the robot hardware and software instructions provided below.

2 Hardware Instructions

The robots are fragile and can easily be damaged if handled improperly. Avoid dropping them, kicking them, or letting them come into contact with water. Make sure the surface on which you are testing is clean (hair and carpet fibers tend to get stuck in the wheels and prevent the robot from moving properly). If testing on a table top make sure the robot does not fall off the end of the table. If you are using the robot at home make sure your pets are not around because they will attack the robot once it starts moving.

Plastic containers are provided for each robot. When not using the robot or when transporting it ALWAYS use the containers to avoid accidental damage to the robot.

If you experience hardware problems contact the TA.

3 Software Instructions

You can download the software package for the robot from the class web page or from <http://www.divent.com>. The package also includes user manual and introduction to BASIC Stamp Programming.

All programs that you write will be written in BASIC Stamp. The language is very easy and you can learn it by looking at the sample source codes provided with the robot.

Before you can use the software you need to connect a the robot to a Windows machine using the provided serial cable. Insert the 9-pin end of the cable into the COM1 serial port of the Windows machine. Connect the 4-pin end to the robot. The connector is located on the bottom of the robot between the two wheels and the black wire on the 4-pin end should point towards the front of the robot.

From MS-DOS prompt or from the run menu in Windows type:

```
stamp2.exe /1
```

This opens an editor window in which you can type your code. The /1 parameter means that your serial cable is connected to COM1. For COM2 use /2. There are very few editor commands that you can execute- press F1 to learn about them.

Now you can turn the robot power on. **IMPORTANT:** The last program downloaded to the robot stays in its memory even after the power is turned off. Be careful if you don't know what that program does because the robot may start moving.

To test the link between the computer and the robot press "ALT I". If you get "Hardware Not Found" error message check the serial cable connection and make sure the robot is powered on and try again.

To load an existing source file press "ALT L". You can try some of the source files provided with the robot (these are the ones with BS2 extensions).

To run the code loaded/typed in the current editor window press "ALT R". **IMPORTANT:** The code is downloaded to the robot in a few seconds. What happens next depends entirely on your code. If you instructed the robot to move it will move immediately. Therefore, it is important to make sure that the

robot is not in a position to fall from a raised surface.

TIP: You may find it useful to start all of your programs with a loop waiting for one of the bumpers to get hit. The following code will do that:

```
lbump    var    in5    'left bumper input

init:
    dirs =0
    pause 100                ' wait for PIC startup
    serout 15\14, 0, ["i"]    ' Initialize PIC processor

wait_bump:
    pause 1000                ' Wait for 1 second
    if lbump then main_code    ' If bumper hit start main program
goto wait_bump

main_code:
    .....
```

4 Problem I: Forward Movement Accuracy

The forward movement of the robot often depends on the friction of the floor surface as well as the alignment of the robot wheels. This problem asks you to design a methodology for: 1) testing the accuracy of the forward movements of the Descartes robot; and 2) estimating the accuracy of the wheel encoders.

The robots can be instructed to move forward by specifying the following command

```
serout 15\14, 0, ["f", lowbyte, highbyte]
```

According to the manual each increment corresponds to 0.2 cm. Your task is to find out if this is correct.

The robot also has two wheel encoders that provide information about how much the wheels have actually traveled. Your second task is to find a correlation between the the number of encoder ticks and the number of centimeters traveled for different floor surfaces.

What to submit: a description of your methodology, your source code, and two graphs representing your measurements. The x-axis of the first graph should represent the parameter given to the ["f"] command and the y-axis should

represent the distance traveled in centimeters. The x-axis of the second graph should represent the distance traveled in centimeters and the y-axis the number of wheel encoder ticks. Submit additional graphs that show the same relations but for two different floor surfaces. For example, if you did the first experiment on your desk try doing the other two on linoleum and cardboard.

What conclusions can you draw from this experiment? Would you say that the wheel encoders of the Descartes robot are accurate? Predictable?

HINT: Your program should be relatively simple. It need not perform all of these measurements at the same time. You can pick the robot up an position it before every trial and you can run the program multiple times. Because of concurrency issues, commands that read the values of the wheel encoders may return incorrect instantaneous results if the robot is moving. Part of your problem is to find a way to deal with these issues.

5 Problem II: Turning accuracy

The previous problem dealt with forward movements only. Here the focus is on turns. Again your task is to design a methodology for measuring the turning accuracy of the robot. The turns should be performed in place.

The robots can be instructed to turn by specifying the following commands

```
serout 15\14, 0, ["l", degrees]    ' turn left
serout 15\14, 0, ["r", degrees]    ' turn right
```

The relations that are interesting in this case are between the parameters of the ["l"] and ["r"] commands and the actual turn angle. Also, between turn angle and the number of left and right encoder ticks.

What to submit: a description of your methodology, your source code, and four graphs representing your measurements. Repeating the experiments for different floor surfaces is optional in this case.

What conclusions can you draw from this experiment? Are the left and right turns of the Descartes equally accurate? Predictable?

6 Problem III: Putting it all together

Based on your measurements in the previous two problems write a program that makes the robot write the word “Mom” on a piece of paper. The robot has a pen holder located in the center of its body. For best results use a pen that does not require a lot of pressure in order to write with and secure it in the pen holder using a rubber band if necessary.

Because the pen cannot be lifted the letters can be joined as in cursive writing. The font of the letters is up to you as long as the first 'M' is capital, the second 'm' is small, the letters are clearly legible and aligned. The size of the letters is also up to you as long as the result fits on a Letter-sized or Legal-sized sheet of paper.

What to submit: your code, a brief description of your approach, and a piece of paper showing your final result.

HINT: Useful code to look at: DC_FIG_8.BS2 and DC_TEST.BS2.