Institute for Aerospace Studies
UNIVERSITY OF TORONTO

LABORATORY IV

Follow-Up Robot

ROB301 Introduction to Robotics
Fall 2015

1 Introduction

This is the fourth lab of ROB301 Introduction to Robotics. The focus for this week is simple trajectories and path-planning. We will investigate introductory control theory for mobile robotic applications. In-depth analysis of mobile robotics, trajectories, and path-planning will be covered in AER521. Regarding error reduction in mobile robotic applications, the graduate course AER1513 will be available to you. In addition to mandatory attendance during the lab session, Lab IV also requires the demonstration of functionality to the TAs as well as a short written report of observations due on 3 November 2015.

2 Purpose

This lab examines foundational principles of mobile robotic motion. Characterization of a mobile platform, simple path-planning, and the build-up of error are investigated.

3 Equipment & Software

The hardware platform used for ROB301 are LEGO Mindstorms EV3 kits. Included on each workstation is a folder labelled "Lab 3" which contains a copy of these lab instructions as well as a PDF of the step-by-step LEGO assembly instructions for the educator mobile platform. This apparatus must be constructed by at least one member of the team in order to complete the tasks of the day. Approximate build time is 30-60 minutes. Please note that the educator mobile platform instructions are found on pages 1-46 of the PDF. The remainder of the LEGO instructions can be ignored for now.

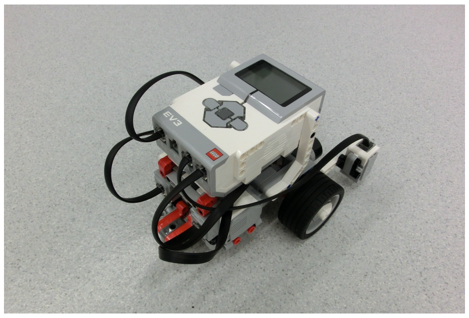

Figure 1: LEGO educator robot

The educator mobile platform consists of two actuators which are two large servo motors for driving.

4 Task Programming

Several path-related experiments will be investigated after first completing a platform characterization to describe its performance capabilities. The first experiment requires the robot to perform the simple yet useful function of wall following. Next, the modified Dubins model with differential steering is to be used as a basis to move from one pose to another. Finally, the robot is to traverse a path consisting of a series of pose-to-pose operations which will allow you to observe the build-up of error.

4.1 Robot Characterization

After construction of the mobile robot platform, you will want to be able to calculate the expected performance of the robot given certain drive inputs. To do this, we first must characterize the physical parameters of the robot. Some relevant measurements are listed in Table 1 with a corresponding Figure 2.

Parameter	Variable	Value	
Wheel Diameter	d	5.5	cm
Wheel Track (inner)	b_{in}	9.0	cm
Wheel Track (outer)	b_{out}	14.5	cm

Table 1: Some important measurements for the mobile robot platform

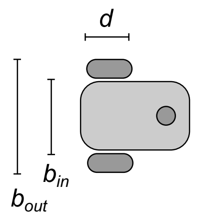

Figure 2: LEGO educator robot

We are primarily interested in how to describe mathematically the motion of the robot which maps well with what physically will occur given specific motor inputs. For example, one full rotation of both wheels in the same direction will cause the robot to travel how far? How many wheel rotations when driven in opposite directions is required to pivot the robot fully in place, i.e., $360°$?

Before Moving On. . . Write a program to verify your calculation of forward movement (for example, three wheel rotations should move the robot how far?). Also write a program to verify how many wheel rotations are required to pivot the robot fully, i.e., $\theta = 360°$. Record your observations and include them in your report. There is no need to demonstrate this to the TA.

4.2 Wall Following

This method of following a path uses known landmarks to navigate. Either the Ultrasonic sensor or the Color sensor could be used in your kit for distance measurement. For now, use the Ultrasonic sensor by mounting it to the robot such that it points out to the side (in-line with the wheel axes). Use a modified version of a P controller (adapted from Lab 3) to keep the robot driving along a wall. For the first 100 cm distance aim for the robot to keep 20 cm away from the wall, for the next 100 cm distance aim for the robot to keep 30 cm away from the wall, and for the last 100 cm distance aim for the robot to keep 20 cm away from the wall.

Before Moving On. . . Write a program to implement the wall-following task for the given distances from the wall in terms of distance travelled. Test your robot along one of the walls in the lab. Record your observations and include them in your report. Demonstrate the functionality to the TA.

4.3 Point to Point

In class, the modified Dubins model with differential steering was introduced as a method to describe the movement of a mobile robot. We will want the robot to go from the start position $[x \quad y \quad \theta]^T = [0 \quad 0 \quad 0°]^T$ to a goal position and orientation $[x_G \quad y_G \quad \theta_G]^T = [200 \text{ cm} \;\; 50 \text{ cm} \;\; 135°]^T$

4.3.1 Straight-Line Path

The simplest method to achieve this movement is to make use of the platform's ability to pivot in place. First, pivot the robot from $0°$ to point toward the goal, travel the straight-line distance to the goal, then pivot the robot to the goal orientation. Work out the required position and orientation of each step. Write a program to complete this task.

4.3.2 Curved Path

A more complicated method assumes that the robot is not capable of pivoting on-the-spot, but must travel in combinations of straight lines and/or smooth curves. Assume that the robot can only turn with a minimum turn radius of 25 cm (larger turns are permitted). Design a path for the robot to begin at the start position and orientation and end at the goal position and orientation. Create a program to complete this task.

Before Moving On. . . Write a program to implement both the Straight-Line Path and the Curved-Path methods to reach the goal position and orientation. Record your observations regarding error and any necessary adjustments to achieve the task and include them in your report. Demonstrate the functionality of both programs to the TA.

4.4 Error Build-Up

Use the Straight-Line Path method from the previous section to drive the robot from the start pose to $\boldsymbol{x}_A$, then to $\boldsymbol{x}_B$, then to $\boldsymbol{x}_C$, then back to the start pose.

$$\boldsymbol{x}_{\text{Start}} = \begin{bmatrix} 0 \\ 0 \\ 0^\circ \end{bmatrix}, \quad \boldsymbol{x}_A = \begin{bmatrix} 100\text{ cm} \\ 0 \\ 90^\circ \end{bmatrix}, \quad \boldsymbol{x}_B = \begin{bmatrix} 100\text{ cm} \\ 100\text{ cm} \\ 180^\circ \end{bmatrix}, \quad \boldsymbol{x}_C = \begin{bmatrix} 0 \\ 100\text{ cm} \\ 270^\circ \end{bmatrix}$$

Write a program to implement the four-point traversal. Make note of your observations on the accuracy of the position and orientation of your robot as it reaches each point. How well did your robot make it back to the start position? What do you think are the largest contributions to the final error? Demonstrate the functionality of both programs to the TA.

5 Written Report

A short written report will be due one week following the lab. No more than five pages, include an introduction, robot characterization discussion, your comments on the wall-following P controller (be sure to include your gain selection!), discussion on the comparison of the straight-line and curved-path point-to-point effectiveness and resulting error, your comments on the error build-up of the square-pattern as well as reasons for the error and suggestions on how to reduce this error, and end with conclusions on your findings. Neatness counts!

DUE: 9H10, 3 NOVEMBER 2015

6 Conclusion

We have introduced some fundamental path-planning concepts and experimented with simple applications.

7 Additional Resources

1. LEGO Mindstorms - `http://mindstorms.lego.com/`
2. Java Programming - `http://www.javaprogrammingforums.com/`
3. leJOS - `http://sourceforge.net/p/lejos/wiki/Home/`
4. leJOS EV3 API - `http://www.lejos.org/ev3/docs/overview-summary.html`