Lab 2: Odometry

CSCI 3302: Introduction to Robotics

Report due Tuesday 2/11/20 @ 11:59pm

2 credit extra per day, up to 7 days/14 credits

The goals of this lab are to:

- Implement the forward kinematics of a differential wheel robot on an embedded system
- To experience and quantify sensor noise from wheel-slip, discretization, and real-time violations
- To understand the concept of "loop closure"

You need:

- A functional Sparkiduino development environment
- A Sparki robot
- A working implementation of a line-following algorithm
- The ArcBotics line following poster

Overview

Odometry integrates the wheel speeds of your robot to calculate its 3D pose (x, y, θ) on the plane. In order to do this, you need to write down the forward kinematics of your robot. As your robot is non-holonomic, you need to calculate the velocity along the robot's x-axis and y-axis, as well as the rotational speed around its z-axis. You then need to transform these velocities into a global coordinate frame and add them up to calculate the robot's pose. You will find out that you cannot do this without error as the robot slips, time discretization introduces inaccuracies, and Sparki's processor might not be fast enough to perform calculations while controlling its motors without delay.

Instructions

Each group must develop their own software implementation and turn in individual lab reports. **You are encouraged to engage with your lab partners for collaborative problem-solving**, but are

expected to turn in your own write-ups. If your group does not finish the implementation by the end of the class, you may continue this lab on your own time as a complete group.

Please submit the following to the course Moodle:

1) A PDF write-up answering each question from Part 4 (please put the number of each question next to your answers, rather than turning in your answers as an essay)
2) The code to your solution.
3) A video of your robot's operation that demonstrates proper odometry calculation and loop closure

Note – The questions in Parts 1-3 are to help with your understanding of the lab. You do not need to answer these in your lab report.

Part 1: Measuring Wheel Speeds

1. Inspect the line following code provided by ArcBotics. What happens during the "delay(100)" statement and where would you want to introduce your odometry code?

2. Think about what would happen if calculating a position update would take longer than 100ms. Think about ways to measure the time the execution of a command takes and making sure every loop takes exactly 100ms. Implement code to reduce the delay statement to make the total time of each loop() call be 100ms *including* your code.
   Hint: check out the millis library for Arduino.

3. Use the millis library to measure how long it takes the robot to move 30cm and calculate its speed from there. This will allow you to calculate its speed in m/s without actually measuring the wheel diameter.
   Hint: The distance from the left-most crease in the folded poster (passing between "Arc" and "Botics") to the beginning of the start line is 30cm

Part 2: Odometry

1. Implement odometry code following Chapter 3 in the textbook (p. 54-55, Eqs. 3.41, 3.42). Calculate the actual distance by multiplying the speeds with the time that the robot actually moves.

N.B. Make sure the robot actually moves for exactly 100ms, not the time of the delay plus the time it takes to execute your code! Initialize your pose (the vector [x, y, θ]) with (0,0,0). Use equation 3.41 to integrate your speeds into positions.

2. Display the robot's pose on its display. (Make sure this operation does not destroy your timing!) What do you expect the display to show when the robot arrives at the start line at the second (the third, the fourth) time? What actually happens?

Part 3: Line Following and Error Mitigation

1. Incorporate code to use the robot's sensors to identify the start line. Implement a *loop closure* mechanism to use this information to prevent your odometry error from growing each lap.

   How could you exploit this information to reduce your error?

Part 4: Lab Report

Create a report that answers each of these questions, in the format:

1. Answer to Q1
2. Answer to Q2

Etc…

1. What happens (in terms of the robot's behavior) during the "delay(100)" statement?
2. What happens if each call to "loop()" takes longer than 100ms? What effect does this have on your position estimation, and why?
3. What is Sparki's average speed (in m/s) when covering the 30cm distance from Part 1?
4. In an ideal world, what should Sparki's pose show each time it crosses the starting line?
5. What does Sparki's pose show after the first lap? Second lap? Third lap? (Without loop closure)
6. How did you implement loop closure in your controller?
7. What are the names of everyone in your lab group?
8. Roughly how much time did you spend programming this lab?
9. Does your implementation work as expected? If not, what problems do you encounter?

To be (optionally) submitted separately via e-mail to Prof. Roncone: aroncone@colorado.edu

(*Optional, confidential, not for credit*) A brief description of any problems (either technical or collaborative) encountered while performing this lab (e.g., issues with the clarity of instructions, clarity of documentation, lab colleague's behavior, etc.)