Lab: The Zip Line

Submission instructions: Students will work for the first time in their design teams (team list posted on MyCourses). Similar to prior labs, this lab consists of three components: demonstration, code submission and lab report. Demonstration instructions can be found in this handout. Lab reports and code submission must follow the guidelines established in this handout and for the course. For more information, see the ECSE211SubmissionInstructions.pdf on MyCourses. Note that as you are now working in design teams, code and report submissions are expected to follow the naming convention Team#_Lab#.zip or Team#_Lab#.pdf as appropriate.

Design objectives

1. Adapt the robot you designed in previous experiments to include a means of traversing a zip line composed of a bent metal rod.
2. Given the coordinates of the zip line platform, design the necessary software to enable your robot to localize and subsequently locate, mount and traverse the zip line.

Design requirements

The following design requirements must be met:

- The system must satisfy the design requirements from Lab 4 with respect to localization, with the following exceptions:
    - You must follow the (**X**,**Y**,**θ**) convention specific in this lab (refer to **Figure 1**).
    - You are free to use either rising or falling edge as desired.
    - You are not required to provide a way of selecting rising or falling edge
    - You do not have to wait for user input after completing ultrasonic or light localization.
- You must provide a suitable user interface using the buttons and LCD display to enter a set of coordinates, ($\mathbf{X_0},\mathbf{Y_0}$) and ($\mathbf{X_C},\mathbf{Y_C}$), with a range of [0,8] and starting corner ranging [0,3].
- Once the parameters are loaded, your program will pause and wait for a button push to begin the localization process.
- After localization, your robot must navigate to the ($\mathbf{X_0},\mathbf{Y_0}$) coordinates specified and align to the zip line axis using ($\mathbf{X_C},\mathbf{Y_C}$).
- After arriving at ($\mathbf{X_0},\mathbf{Y_0}$), your robot must pause and wait for input.
- After receiving input at ($\mathbf{X_0},\mathbf{Y_0}$), your robot must mount and traverse the zip line, dismounting on the other side and coming to a controlled stop.
- Your robot must complete all parts of the task (i.e. arrival at the far end of the zip line) in a maximum of 5 minutes.

Demonstration (30 points)

The design must satisfy the requirements by completing the demonstration outlined below.

Design presentation (5 points)

Before demoing the design, your group will be asked some questions for approximately 5 minutes. You will present your design (hardware and software) and answer questions designed to test your understanding of the lab concepts. Grades will apply to the entire group, although TAs reserve the right to grade individually if they deem it necessary.

You must present your workflow, an overview of the hardware design, and an overview of the software functionality. Visualizing software with graphics such as flow charts is valuable.

Demonstration procedure

The zip line will be set up on an 8x8 grid as shown below in **Figure 1**.

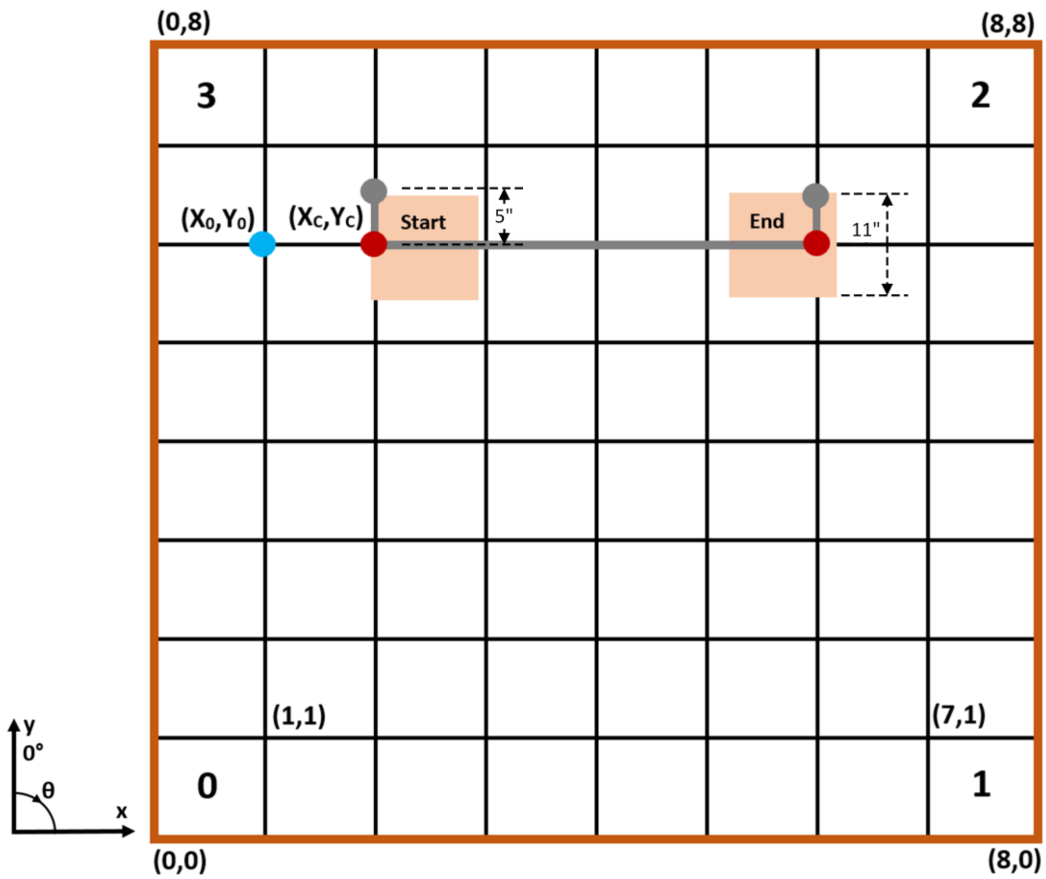

Figure 1. Zip line set up and (X,Y,θ) convention on the 8x8 grid. Measurements are in inches.

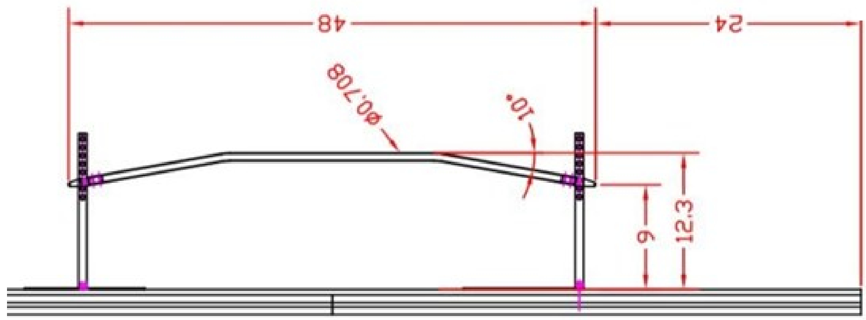

Figure 2. Side-view of the zip line. Measurements are in inches.

The zip line is set up such that it aligns with the grid as shown in the two figures (aligned with the x-axis). At the start of the demo, a TA will provide you with the following parameters as labeled in **Figure 1**. Note that the (**X**,**Y**,**θ**) convention is fixed according to **Figure 1** in this lab.

- ($\mathbf{X_0},\mathbf{Y_0}$): grid crossing coordinate 1-tile away from the zip line, indicated by the blue dot
- ($\mathbf{X_C},\mathbf{Y_C}$): center coordinate of the zipline's starting point, indicated by the red dot
- **SC**: starting corner number ranging from [0,3]

For a successful demonstration, the following steps must be followed:

- Enter the parameters given to you, i.e. ($\mathbf{X_0},\mathbf{Y_0}$), ($\mathbf{X_C},\mathbf{Y_C}$), **SC**.
- Place your robot in a starting corner according to **SC**.
- Start your robot: it should then proceed to localize, navigate to ($\mathbf{X_0},\mathbf{Y_0}$) and align itself with the zip line axis using ($\mathbf{X_C},\mathbf{Y_C}$).
- Record the pose (**X**,**Y**,**θ**) at this location once your robot has paused.
- Press a button to continue: your robot should then proceed to mount the zip line and traverse to the other side. It is important that you dismount the zip line or land in a controlled fashion, e.g. slow down as you reach the end.

Pose Orientation Accuracy (2.5 points)

2.5 points are given for orienting the robot parallel to the zip line axis to within an error tolerance of ±2.5°. No points are given for angles beyond ±2.5°. Hence the following points are awarded for the robot's actual orientation:

- **± [0, 2.5] °** → **2.5 points**
- **± (2.5, ∞) °** → **0 points**

Pose Position Accuracy (2.5 points)

2.5 points are awarded for reaching point ($\mathbf{X_0},\mathbf{Y_0}$) to within an error tolerance of 2 cm using a Euclidean distance measure. No points are given for errors beyond 2 cm. Hence the following points are awarded for the actual robot's position relative to ($\mathbf{X_0},\mathbf{Y_0}$):

- **[0, 2] cm** → **2.5 points**
- **(2, ∞) cm** → **0 points**

Zip Line Traversal (20 points)

- **10 points** are awarded for successfully mounting the zip line as well as rolling at least 10 cm along the rail.
- **5 points** are awarded for a complete traversal of the zip line.
- **5 points** are awarded for a successful, controlled dismount where your robot must be oriented parallel to the zip line axis within an error tolerance of ±30°.

Provided materials

Sample code

No sample code is provided for this lab.

Physical material

A zip line prototype mounted on an 8x8 grid floor will be provided for this lab.

Implementation instructions

The implementation of this lab is at your discretion. Since this lab will be done in conjunction with the entire design team, you have the opportunity to explore different solution approaches.

Report Requirements

The following sections must be included in your report. Answer all questions in the lab report and copy them into your report. For more information, refer to ECSE211SubmissionInstructions.pdf. Always provide justifications and explanations for all your answers.

Section 1: Design Evaluation

You should concisely explain the overall design of your software and your robot (hardware). You must present your workflow, an overview of the hardware design, and an overview of the software functionality. Visualizing software with graphics is valuable.

Section 2: Test Data

This section describes what data must be collected to evaluate your design requirements. Collect the data using the methodology described below and present it in your report.

Pose accuracy (*10 independent trials*)

1. Place the robot in corner 0 along the 45° line as in Lab 4, randomly oriented each time.
2. Localize to (1,1) and 0°.
3. Navigate to a known tile position (6,2) on the map and orient to 270°.
4. Measure the Euclidean distance error and the angle error with respect to (6,2) and 270°.
5. Report in a table: Euclidean distance error and final angle error, for each trial.

Mounting the zip line

1. Assume that **θ**, the angle error, is 0°.
2. Place the robot at ($\mathbf{X_0}$,$\mathbf{Y_0}$) as shown in **Figure 1**.
3. Set the robot such that there is an angle error **θ** between the robot's orientation and zip line.
4. Instruct the robot to drive forward and mount the zip line.
5. Traverse the zip line for about 10 cm.
6. If step 5 was successful, go back to step 2 and increase **θ** by a specific amount, e.g. 2°.
7. If step 5 was unsuccessful, report the last **θ** when the robot successfully mounted the zip line. Note that the robot should not mount for any angle error more than the last **θ**.
8. Report in a table: each value of **θ** tested and whether or not the robot was successful.

Landing accuracy (*10 independent trials*)

1. Place the robot at ($\mathbf{X_0}$,$\mathbf{Y_0}$) and orient at 0° as shown in **Figure 1**.
2. Instruct the robot to drive forward and mount the zip line.
3. Traverse the zip line.
4. After dismounting the zip line, note the landing angle error $\boldsymbol{\theta_L}$ with respect to the zip line.
5. Report in a table: the landing angle error $\boldsymbol{\theta_L}$ for each trial.

Section 3: Test Analysis

1. **Pose accuracy**: compute the mean and standard deviation of the Euclidean distance error and final angle error.
2. **Mounting the zip line**: list the angle errors tested and if the mounting was successful.
3. **Landing accuracy**: compute the mean and standard deviation of $\boldsymbol{\theta}_L$.
4. Be sure to show general formulas and sample calculations.

Section 4: Observations and Conclusions

1. How accurate does your system (i.e. localization, navigation, odometry) need to be in order to successfully mount the zip line? Refer to your collected test data.
2. Discuss any problems your robot has when mounting and traversing the zip line. Does it perform consistently? You may wish to discuss topics related to your mechanical design such as whether or not the wheels are level with respect to the ground, your mounting success rate and the weight distribution.
3. What factors contributed to the robot's performance (i.e. pose, mount, dismount) in your tests?

Section 5: Further Improvements

1. Propose a way to refine your pose accuracy before mounting the zip line.
2. Propose a hardware or software improvement to improve the robot's traversal or dismount from the zip line.
3. Propose a way to correct your odometry after landing from the zip line.

Frequently asked questions (FAQ)

1. **Can we demo the zip line part separately (i.e. mounting, traversal, dismounting) if our pose is inaccurate?**
   No, both parts must be performed in succession.

2. **Do we have to follow the (X,Y,θ) convention specified in Figure 1?**
   Yes. Also note that the bottom-left corner of **Figure 1** is specified as (0,0).