Exercise 6: Field data acquisition and analysis

Rahul Rajendra Pai
rahul.pai@chalmers.se

Pierluigi Olleja
pierluigi.olleja@chalmers.se

Tianyou Li
tianyou.li@chalmers.se

October 2, 2023

Introduction

In this assignment you will be using your knowledge on safety measures and signal processing to analyze real-world data collected in the field. The aim of this assignment is to investigate how vulnerable road users (VRUs) interact. This assignment consists of two parts. In the first part, you will be collecting data with the LIDAR. In the second part you will analyze the data you collected to model VRUs interactions. In this assignment you will be using MATLAB.

Note: You are encouraged to discuss your solutions with others but you must write and submit your own code implementation and answers.

Deadline: October 16, 2023 (23:59)

Learning objectives

After having performed this assignment, you shall be able to:

- Collect real-world data in real world from a LIDAR based system
- Process and filter LIDAR data
- Calculate safety indicators to create a simple model for the interaction between VRUs
- Select parameters to design a collision warning system

Preparations

1. Download the material from Canvas. `Exercise.m` is the template script to solve the second part of the assignment. The script contains some guidelines to complete the exercise. The file `playLidar.m` is the toolkit to visualize the collected data. The file `bag2csv.m` contains a function to convert the

file that contains the LIDAR measurement (.bag) into a comma separated values (.csv) file that only contains LIDAR data. The file `importLidarData.m` contains a function to import the .csv data into MATLAB.

2. You have the choice to collect your own data on campus (see info on Canvas), or to work with the data from a previous year (`2018-10-24-11-11-25_0.bag`).

3. We are using the Robot Operating System (ROS) to record the LIDAR data with our data logging platform[1]. To be able to read the recorded data in MATLAB, you need to have the ROS Toolbox[2] installed.

Tasks

Part A - Data collection

Instructions to collect the data will be given in class. The instructions will include the description of the LIDAR-based platform and how to operate it. You will be outside with the teaching assistants to collect LIDAR data using a handheld LIDAR device. By holding the LIDAR in front of you and approaching pedestrians (this will later on be referred to as the 'event of interest'), you will record data which will be used in the second part to analyze pedestrian interaction.

Note: Due to the pandemic, this part is voluntary in this year's round of the course. If you do not wish to collect your own data, you can proceed to Part B with the data from the previous round (`2018-10-24-11-11-25_0.bag`).

Part B - Data analysis

1. The Lidar has a field of view of about 190 degrees and a range of about 120 meters. At each timestamp, the Lidar scans the environment and collect 1521 measurements equally spaced between -95 deg and +95 deg. Figure 1 below shows the reference system for interpreting the data.

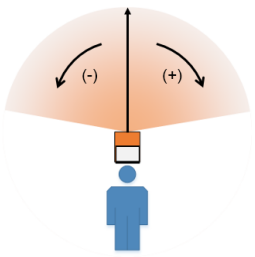

Figure 1: LIDAR reference system

[1] `https://github.com/ruvigroup/div_datalogger`
[2] `https://se.mathworks.com/products/ros.html`

2. The LIDAR saves the data in the ROS .bag format. Save this file into the folder of the exercise. The .bag file needs to be converted into a .csv file via the function `bag2csv` which extracts only the LIDAR data from the bag file. That reduces file size, in case you have video data in your collected data (which is not the case for the given example recording). The function takes as input the relative path of the .bag file to be converted, e.g. `bag2csv('2018-10-24-11-11-25_0.bag')`. When you run `bag2csv`, a new .csv file will be saved in the same location of the .bag file. This .csv file is the one you should submit in the end.

3. With the function `playLidar`, you can visualize the ROS .bag file you recorded (both LIDAR and video data). When you run the script without any input, a dialog box appears. Select the .bag file that contains the measurements. Otherwise, you can run the function specifying the .bag file to be loaded as a input to the function, e.g. `playLidar('2018-10-24-11-11-25_0.bag')`. The interface shows both the timestamp in seconds and the frame number. Working with frame numbers makes it easier to analyze the data in MATLAB (the frame number corresponds to the row number in the .bag file). Scroll through the frames (you can also use the arrow keys, left/right = +/- 1 frame, up/down = +/- 10 frames) and identify the frames that contain relevant events (in particular, the start and the end frame for each event). You may also want to limit the field of view. A narrow field of view may help to differentiate among targets.

4. Once you have identified the events of interest and you found the optimal field of view settings, you can start coding in the main script `Exercise.m`. Before anything, set the filename of the .csv you will be using for your analysis. The script will import the .csv file into a MATLAB structure data.

5. To get acquainted with processing LIDAR data, plot the LIDAR measurement at a single timestamp. Plot the measurements in the whole field of view of the LIDAR, highlight the data within the chosen (bounded) field of view, and mark the closest point in the field of view (which is a reasonable approximation of the obstacles). The LIDAR data are collected as polar coordinates. Thus, the data need to be transformed to be able to plot a graph in Cartesian coordinates (try the MATLAB function `pol2cart`). Try to replicate Figure 2:

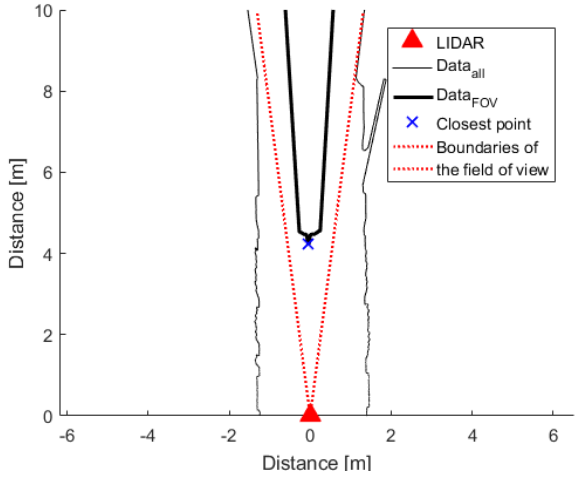

Figure 2: LIDAR visualization via the `playLidar` function.

6. In a single figure, plot the distance measurements for all the events of interest. You should obtain something like Figure 3. Data can be noisy. If so, apply a filter to the data to reduce the noise and remove artifacts. Try, for example, the MATLAB function smooth.

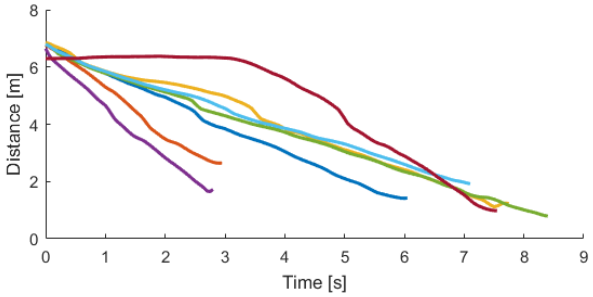

Figure 3: Distance over time, obtained from all events of interest.

7. In a single figure, plot the relative speed (range rate) with the obstacles for all the events of interest. You should obtain something like Figure 4. Data can be noisy. If so, apply a filter to the data to reduce the noise and remove artifacts.

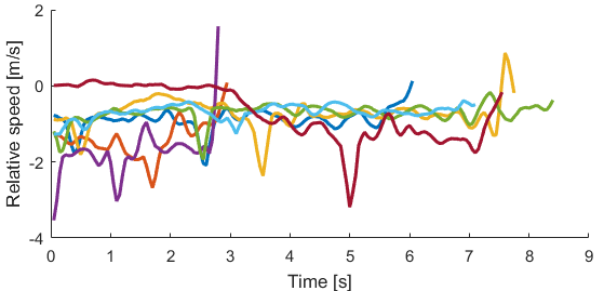

Figure 4: Relative speed over time, obtained from all events of interest.

8. In a single figure, plot time to collision (TTC) with the obstacles for all the events of interest. You should obtain something like Figure 5. Data can be noisy. If so, apply a filter to the data to reduce the noise and remove artifacts.

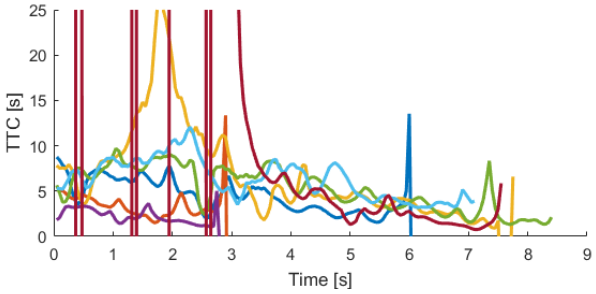

Figure 5: Time-to-collision (TTC) over time, obtained from all events of interest.

9. Finally, answer the question: What safety measure would you use to design a warning that alert the user that is about to collide with an obstacle? You may want to use one of the safety measures you computed in the script of find a more accurate one. What value of such measure would you use to trigger a warning? **Write your answer in the MATLAB script and enclose it in a comment.**

Submission

Submit your solutions in the assignment in Canvas. Submit the `Exercise.m` script together with your recorded .csv file (only if you recorded your own data). If the file is too large, upload it e.g. using Chalmers box (`https://chalmersuniversity.app.box.com/`), and submit the link to the file in a comment. It is sufficient if at least one group member submits your solution in Canvas. See deadline above.