Midterm Project 1: Scan Matching

Instructions: This project can be completed in teams consisting of up to two individuals. The team will need to generate a prediction of the sequences of poses that a robot traveled in corresponding to 2D laser scans provided in a file `sims_N_scans.ext` where `N` is a number and the extension is either `.npz` or `.mat`.

1.1 Background

LiDAR Scan

Figure 1: A relatively inexpensive 2D LiDAR (example).

A 2D LiDAR sensor typically uses a rotating mirror or sensor to emit laser pulses across a single, fixed angle to capture objects in its surroundings. This sensor or mirror rotates within a horizontal or vertical plane to scan across that plane. Given a frame coincident with the center of rotation, the LiDAR sensor returns a sequence of points in polar coordinates (r_i, ϕ_i), which are often collected together into a single scan. These points correspond to the nearest light-reflecting object along the direction defined by ϕ_i in the sensor's frame of reference. The scans can also be stored using Cartesian coordinates (x_i, y_i) where

$$x_i = r_i \cos \phi_i, \text{and}$$
$$y_i = r_i \sin \phi_i.$$

The data provided are in Cartesian coordinates with units in meters.

Scan Matching

When a robot moves through the world, the LiDAR scanner moves with it, capturing distances to objects along directions. If the motion is small, then that motion can be estimated from two laser scans by trying to 'align' or 'match' the scans. The idea is that by transforming all scans into a common frame, points in scans corresponding to the

same physical object should be located at the same point. Equivalently, 'walls' seen in two nearby scans should lie on top of one another when the right transformation is estimated.

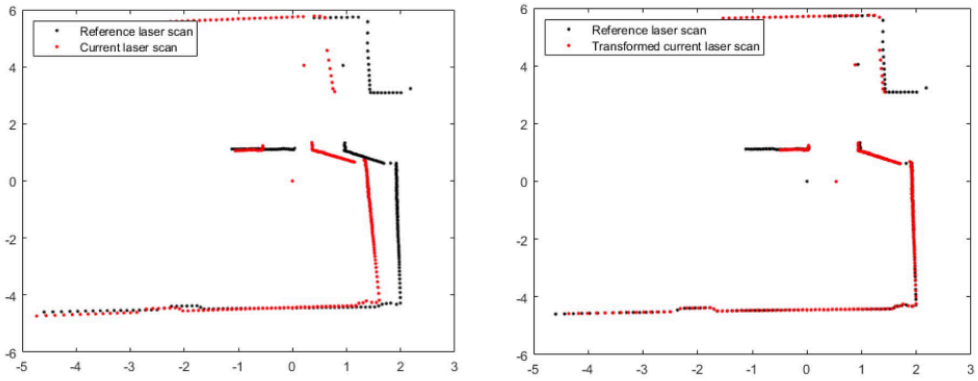

Figure 2: We can plot the scans as seen by the robot in its own frame when it is in two different locations (left). When we correctly estimate the relative motion and apply it to one of the scans, we bring the scans into a single consistent reference frame, and objects and walls overlap (right).

By successfully matching pairs of scans, the long-term motion of the robot can be estimated just from the laser scans obtained in a sequence of (reasonably close) positions in the real world.

Modeling

The pose of the robot in a reference world frame can be described by three numbers (X_i, Y_i, θ_i), which also describe the location of a frame that moves with the robot. The pair (X_i, Y_i) is the location of the center of the robot (origin of robot frame, same as sensor frame) in the world frame, and θ_i is the angle in which the robot-centric direction $\phi = 0$ lies. When the robot frame coincides with the world frame, the robot pose is $(0, 0, 0)$. If we obtain a laser scan consisting of points $\{x_j, y_j\}$ in the frame of the robot when the robot is at pose (X_i, Y_i, θ_i), then these points are located at $\{X_i + x_j \cos\theta_i - y_j \sin\theta_i, Y_i + x_j \sin\theta_i + y_j \cos\theta_i\}$ in the world frame. Another way of looking at this transformation is that points in a frame were moved in unison to a new location determined by the pose (X_i, Y_i, θ_i).

For two scans to match, we can temporarily treat one as the reference frame, and try to find the numbers (X_i, Y_i, θ_i) that would 'correctly' transform the scan of the second frame to align with the scan of the reference frame. Therefore, the goal is to find the numbers (X_i, Y_i, θ_i) that provide the **best** transformation.

1.2 Problem

The goal of the project is to use optimization to find the best transformations that would align the scans. Your team must create code that can estimate the sequence of poses

$\{X_i, Y_i, \theta_i\}$ corresponding to a sequence of scans provided in a file as mentioned in the instructions. The estimate is to be provided as a 2D plot of the sequence of points corresponding to (X_i, Y_i) in the plane, and a line through these points indicating θ_i. Example scans are provided during the project period to help develop and evaluate your algorithms.

To approach the project, you will need:

- An evaluation function that provides a scalar value for how good (or bad) a set of values (X_i, Y_i, θ_i) is as a candidate for the transformation between two scans.
- An algorithm to minimize this function, given two scans.
- A method to convert relative transformations between pairs of scans into an overall path of the mobile robot.

1.3 Evaluation

The evaluation is based on a report that contains:

a) Description of all evaluation functions and algorithms you considered.
b) A discussion of what changes you made along the way and why.
c) A discussion about whether the changes had the impact you expected or not.
d) A summary of the performance of the final algorithm at the end of the project period. This performance is measured in terms of
   - average run time per iteration,
   - number of iterations per scan match,
   - average error in scan match, and
   - average error in pose.
e) A discussion of what steps could be taken, if any, to improve the final performance measures

1.4 Hints

- It is not expected that you will obtain an accurate prediction of the robot's motion.
- To prepare the report, it is a good idea to log all decisions and save salient plots (objective vs iteration, scan match results, etc) as you go, instead of waiting to compile everything at the end.
- The emphasis is on connecting theory and practice during the project
  - Try simple things first and use the results (and surprises) to inform choices
  - An approach that produces highly accurate predictions without explainable choices is not success
- Two popular methods for scan matching involve the Iterative Closes Point algorithm, and the Normal Distribution Transform
  - Copying from existing implementations of these algorithms that produce highly accurate predictions is not success

- Test on synthetic 'ideal' data for which you know how your optimization algorithm should perform
  - Example: If one scan has points (x_i, y_i), then create the scans $(x_i', y_i') = (x_i + \Delta x^\star, y_i)$ or $(x_i', y_i') = (x_i \cos\theta^\star - y_i \sin\theta^\star, x_i \sin\theta^\star + y_i \cos\theta^\star)$. A point seen in one scan is also seen in the other, so you can use a loss based on the sum of errors between all pairs (x_i', y_i') and (x_i, y_i) transformed by $(\Delta x, \Delta y, \theta)$. For realistic scans, the same physical point is mostly not observed in two scans.
- Create reusable tools for visualizing and plotting early on to help check and debug results