ESE 650
Uriah Baalke
SLAM

The purpose of this project is, given lidar, imu, and encoder data from a differential drive robot, implement a SLAM algorithm that, with reasonable robustness, can construct a map from a given robot trajectory.

One common method to achieve this is to implement a particle filter, which represents our belief of the robot state over a set of particles. For this implementation at each step each particle is weighted according to how well the current lidar measurement matches the map belief. With a uni-modal (or single) map, the map is updated based on the best particle. When a large enough number of particles contain very low weights, and a small enough number of particles contain large weights, then a re-sampling step is undertaken, which takes low weight particles and re-samples them such that they are nearer high weight particles.

Since we have chosen a unimodal belief one alternative to the particle filter is to scan-match by looking at a discrete set of states around the belief state. This is the approach I have taken so far, and attained reasonable results. At each step, predict the state given odometry data, then for a set of angles near the current angle and a set of positions near the current state the most recent laser scan is compared to the current map belief and a value for each sampled state is acquired. Once a set of weights is attained the state is updated using the error between the best sample and the current state. With correct tuning this works very well as with the particle filter, performance is improved by sampling more states and tuning beliefs.

A set of maps are provided showing the difference between raw odometry and the scan matching approach. From the figures it is clear that scan matching with a small sampling region is a significant improvement over raw odometry and with a larger sampling region (90 samples) very precise maps are possible.

Thinking about relating this to my final project, I am thinking about projects like kinematic model identification using scan matching, dynamic allocation of resources for scan matching based on model confidence, and graph based slam using scan matching for the front-end of the algorithm.

Figure 1: Test #1 with scan matching

Figure 2: Test #1 No Matching

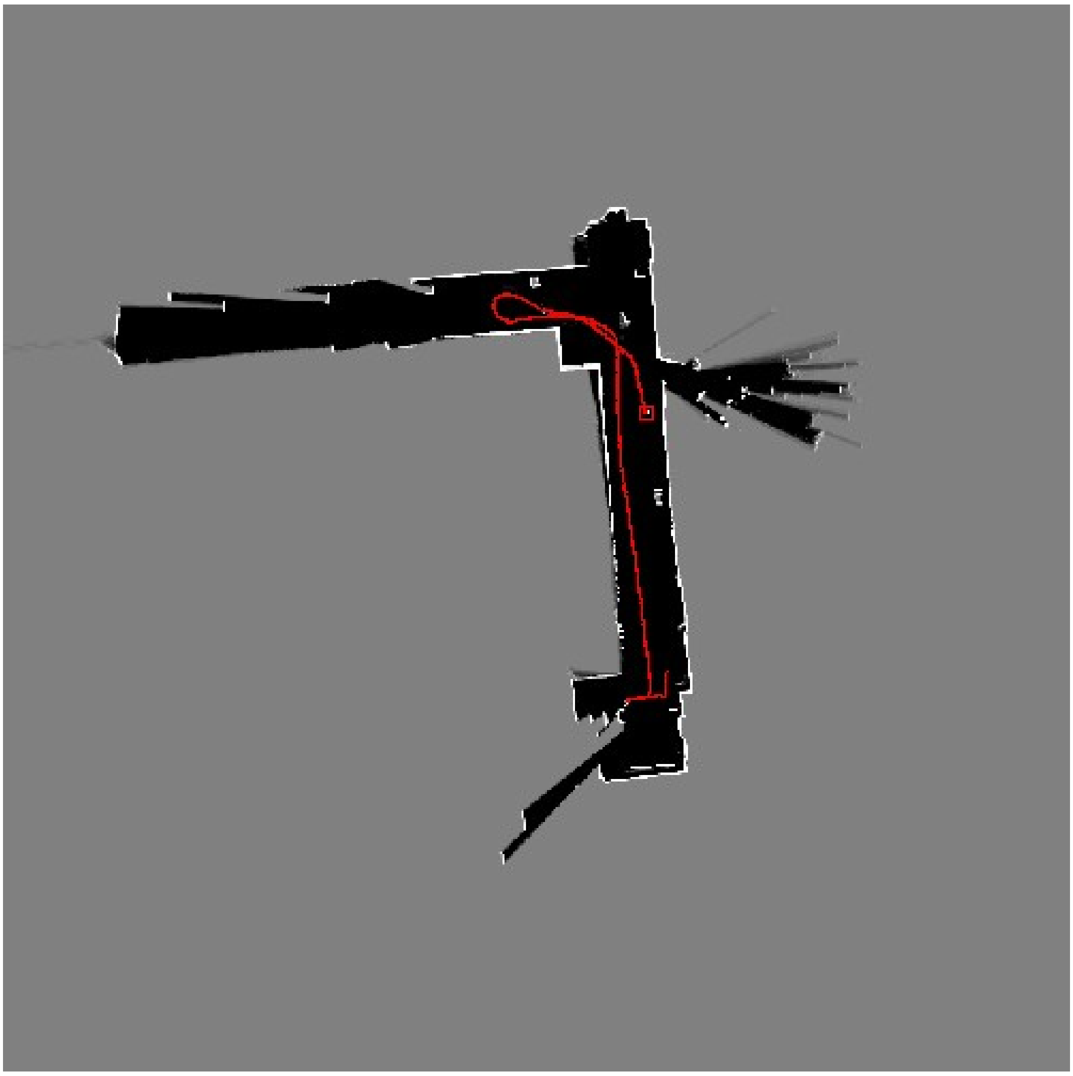

Figure 3: Test #1 90 samples (3x3 dx-dy, 5 dtheta samples over 0.1 rad region)

Figure 4: Test #2 Raw Odometry

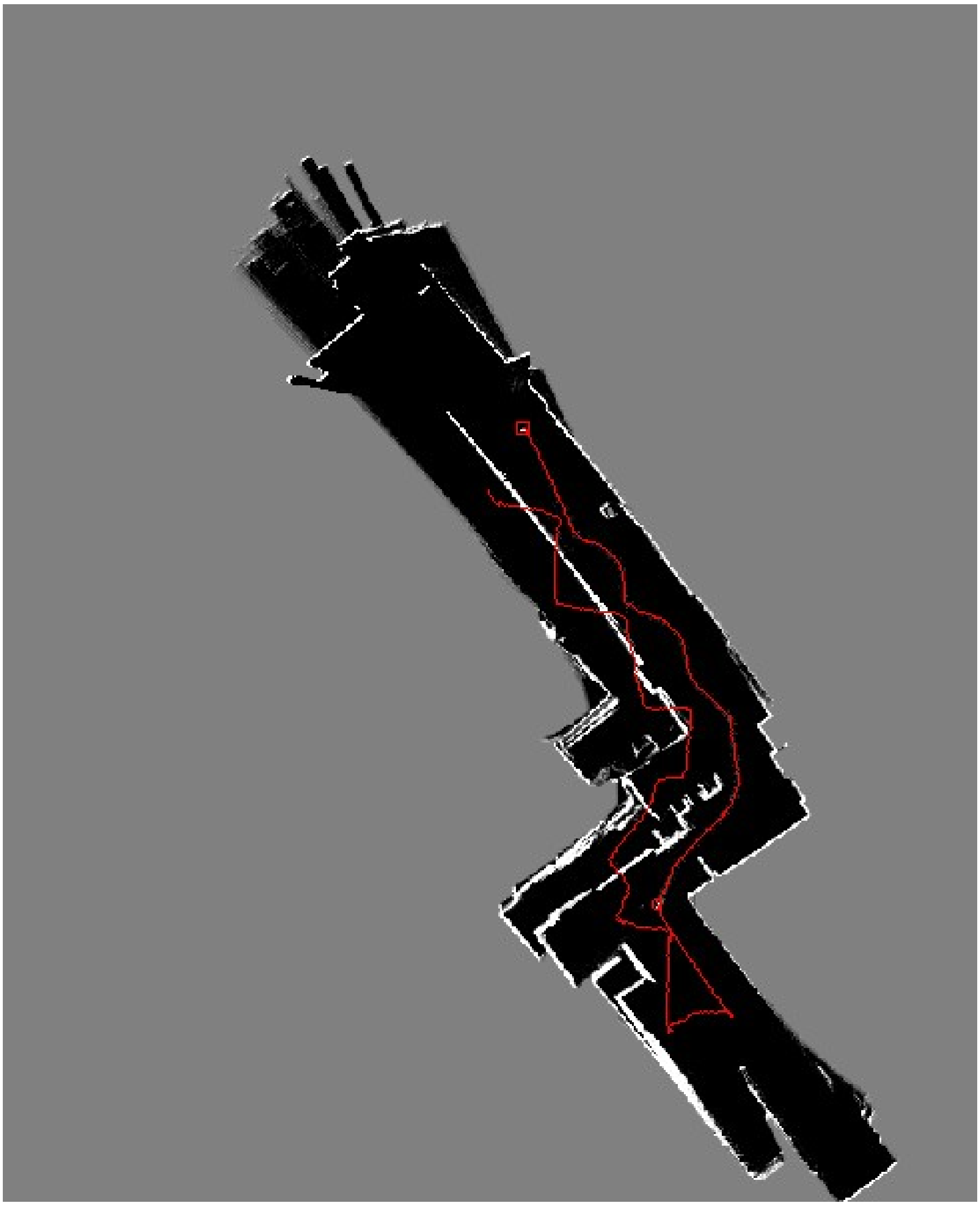

Figure 5: Test #2 With Scan Matching

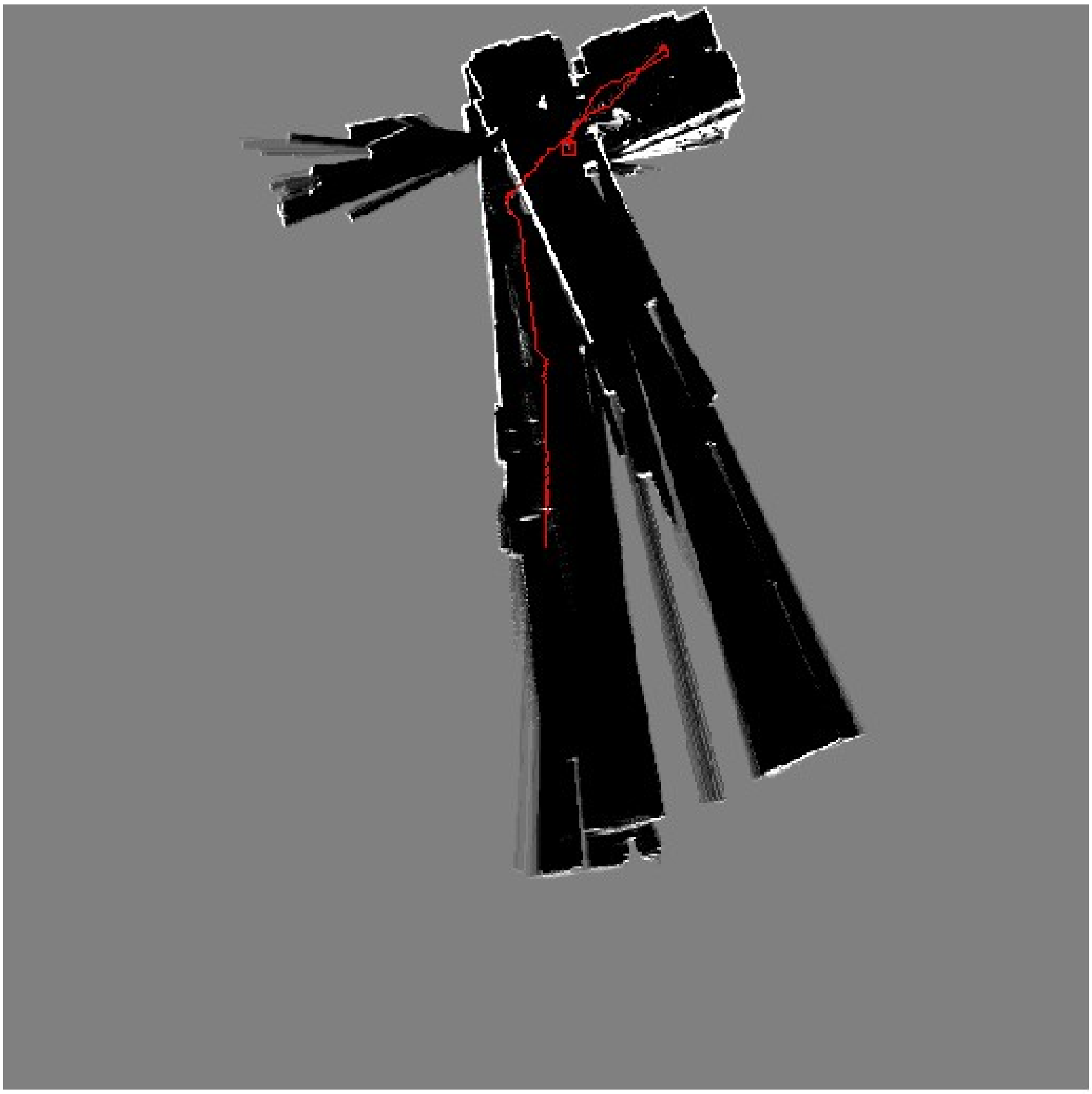

Figure 6: Test #3 Scan matching small number of samples

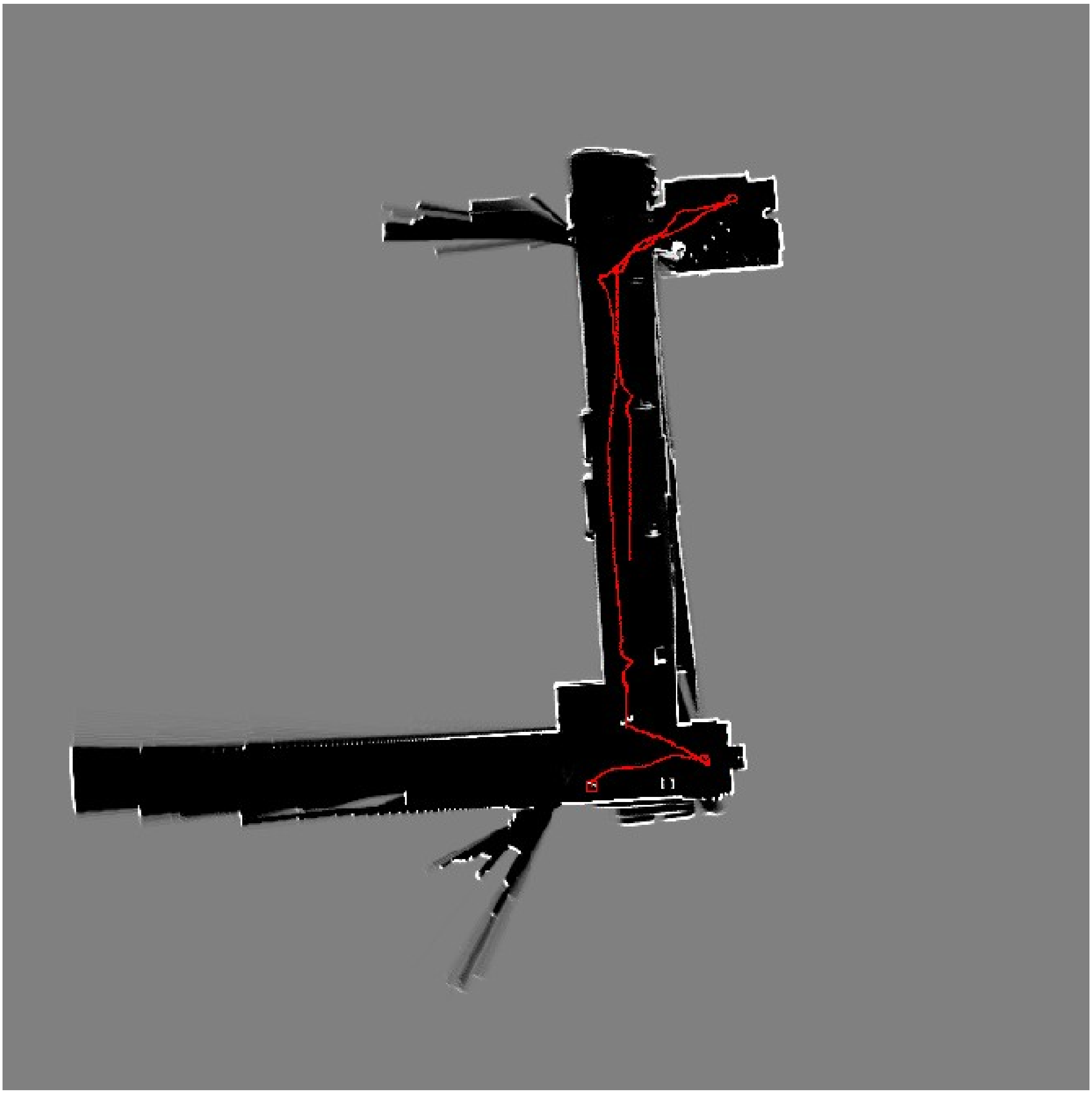

Figure 7: Test #3 with 90 samples (3x3 dx-dy, 5 dtheta samples over 0.1 rad region)

Figure 8: Test #3 Raw Odometry