AERSP 497 Autonomy

Project 2

Name	Contribution	Contribution %	Honor Statement Sign
Nicholas Giampetro	Code & Report	33	NG
Jackson Fezell	Code & Report	33	JF
Ankit Gupta	Code & Report	34	AG

Introduction

This project is an exploration and simulation of the least-squares method to solve the Simultaneous Localization and Mapping (SLAM) problem in two dimensions. There were four MATLAB data sets provided to run this method on: simulation-pose-pose, intel, simulation-pose-landmark, and dlr. The expected results and error values for these data sets are shown below.

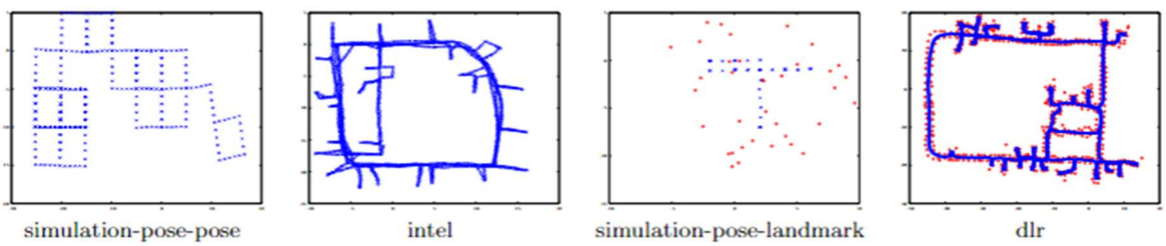

Figure 1: Result for each dataset.

dataset	initial error	final error
simulation-pose-pose.dat	138862234	8269
intel.dat	1795139	360
simulation-pose-landmark.dat	3030	474
dlr.dat	369655336	56860

As the acronym suggests, SLAM is used to track the movement of an Autonomous Vehicle (AV) while also creating an image of its surrounding environment at the same time. Without access to prior information, the AV can understand where it is relative to its local environment, typically becoming more accurate in its estimation as the number of measurements taken increases. The SLAM problem can also be used by multiple AVs at once to move throughout a terrain by relaying mapping and positioning information to the other AVs.

This information is collected through measurement devices, such as, Light Detection and Ranging (LiDAR) and Inertial Measurement Units (IMUs), and then processed. The processing of the collected data is affected by the confidence of the correctness of the raw data within the least-squares method.

The least-squares method is a technique that applies a regression analysis to approximate the state of an autonomous system and its environment. This method works because the system is overdetermined, meaning there are more equations than unknowns. By reducing the residual difference between observed and fitted values, a good approximation of the environment can be found to solve the SLAM problem in a short period of time.

Results

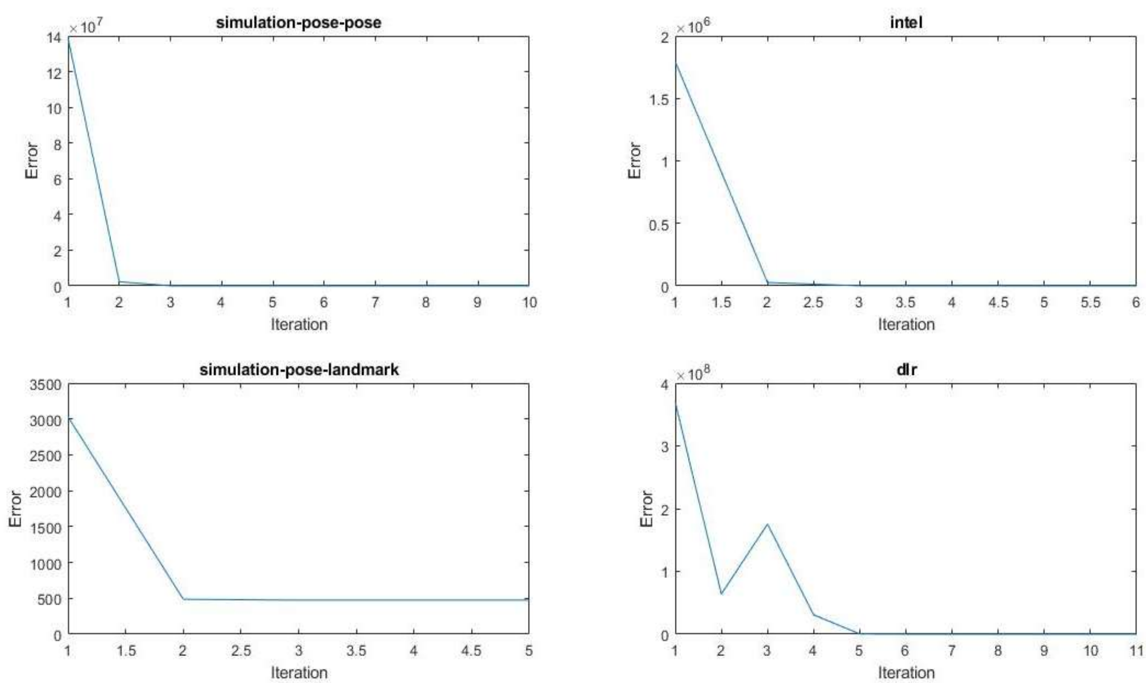

Figure 1: The calculated error of each simulation through the number of iterations each took to complete.

Starting with the error values calculated with equations (4) and (14) in the handout, it can be observed that as the least squares method iterates, the error decreases. This means that the algorithm is creating a two-dimensional representation of the real environment for the autonomous system. It can also be observed that for these cases, the least squares method can converge on a viable solution within only a few iterations which would save computational resources on an autonomous system.

These error values also match with the provided correct error values in the handout which indicates that the MATLAB code being used to generate these plots is executing the least squares method correctly.

Armed with the understanding that this method does result in a good approximation of the environment, a greater understanding of SLAM can be obtained from investigating further details of one of the provided test cases. The plots generated by the MATLAB code shown below show the refinement of the estimated state of the environment over several iterations of the intel case.

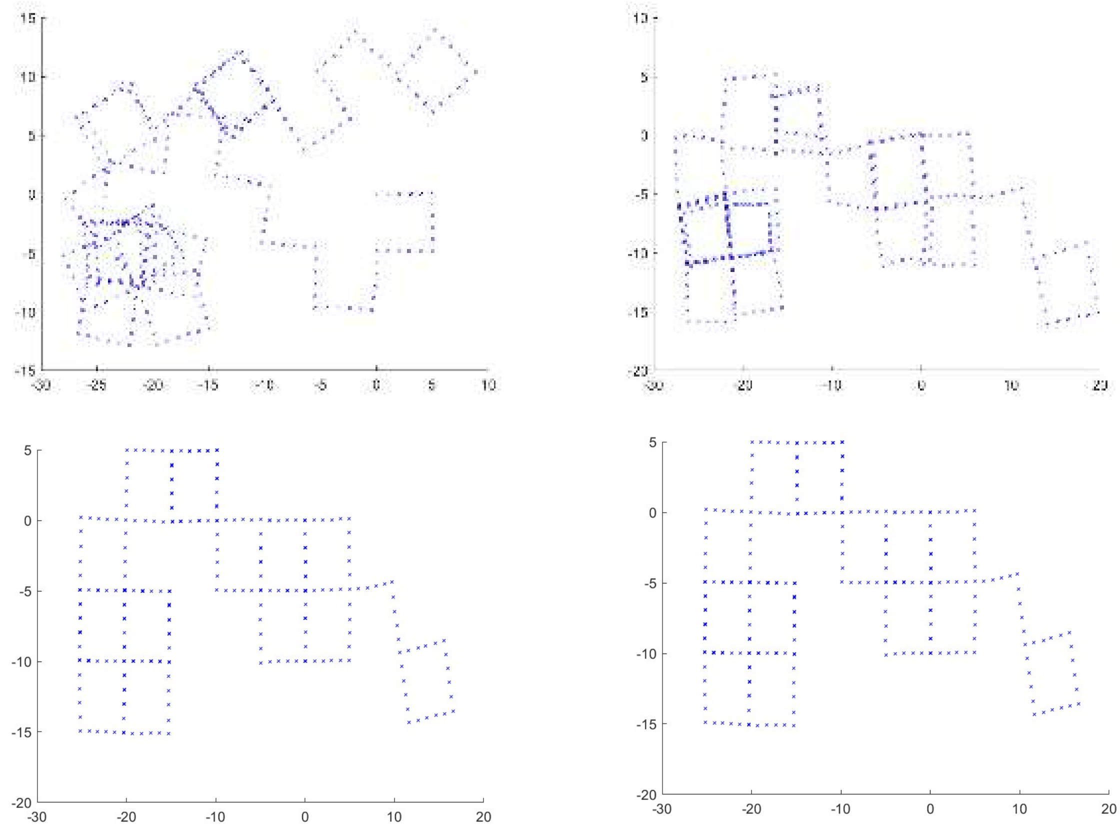

Figure 2: From left to right, top to bottom are the first, second, fifth, and tenth (final) iteration of the drone's pose as it navigated its environment.

As the AV wanders through its environment, it will track its own position and orientation – pose – to know where it is relative to the map. Figure 2 depicts the process of the AV iterating through the least-squares method to minimize, and therefore optimize, the error contained within the graph. The first iteration shows a significant amount of error, which is depicted by Figure 1, with the majority of it visibly stemming from the measured orientation. After the second iteration, the graph greatly and quickly improves. By the fifth iteration, the pose graph looks visibly identical to the final iteration. The algorithm gains a higher confidence in its ability to map the terrain as the AV continues to move around and recognize where certain features of the map overlap. The final iteration depicts the path the AV took to collect its data with relatively high accuracy.

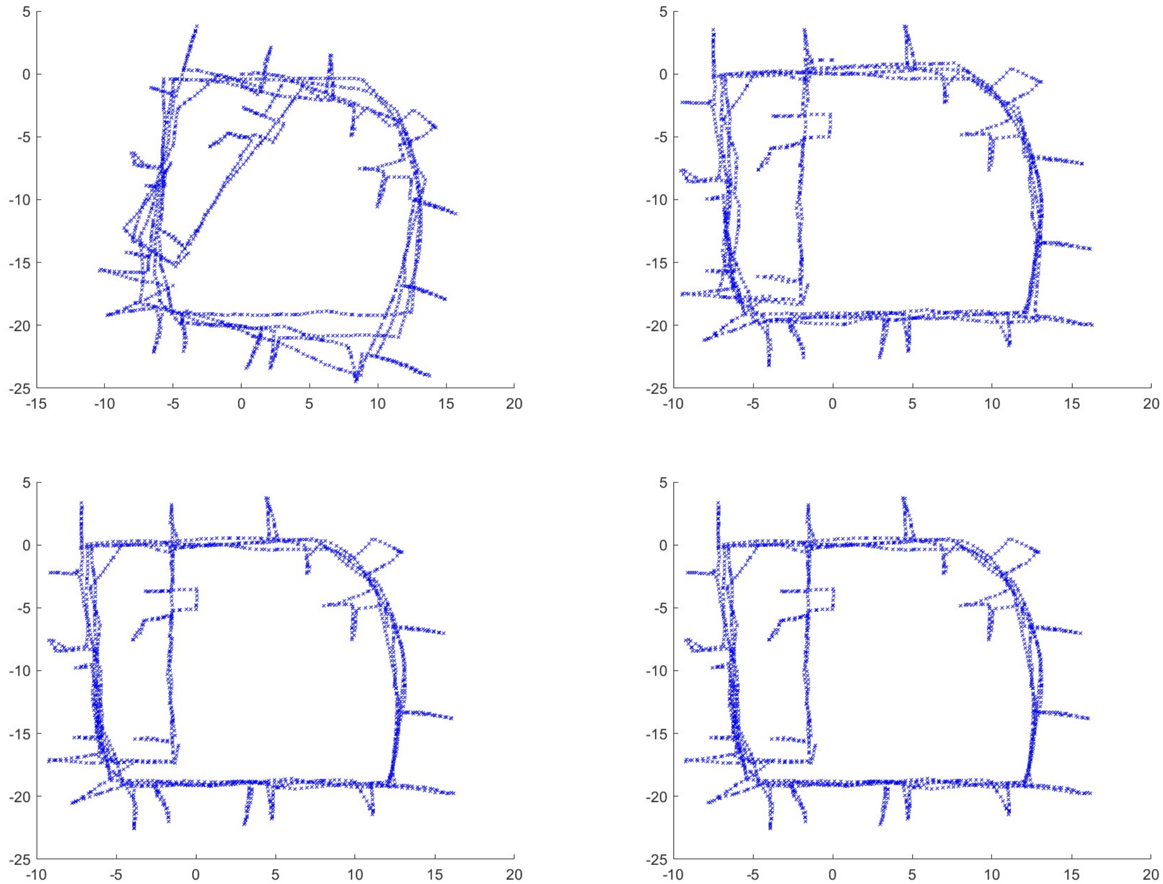

Figure 3: From left to right, top to bottom are the first, second, fourth, and sixth (final) iteration of the intel graph using the drone's pose during navigation.

Looking into the intel case, it can be observed that as the least-squares method iterates, the plot converges onto a single image utilizing pose-pose strategy, and by the second iteration there is very little visual difference from the final iteration to the human eye. By the third iteration the map is even more like the final iteration to the point that it is functionally identical to human observations. This visual observation is backed up by error plot of intel shown in Figure 1, where to a human observation the third iteration has a near identical error value to the last iteration of intel.

The plot also closely matched the provided result of intel shown in the handout, further indicating the validity of the MATLAB code's ability to perform SLAM.

As for the pose-landmark and dlr case, the difference is that AV is mapping only its surroundings using landmarks to identify certain parts of the surroundings. Since what is being mapped is different, the error of the edges between points is calculated using a different error function. This leads to different errors and solving speeds in the optimization algorithm. The pose-landmark error started out three to five magnitudes smaller than all the other algorithms, but still leveled out in the same number of iterations as the pose-pose and intel case. The dlr error took the longest to settle out of the four, taking a total of 5 iterations before stabilizing around the same value. This is most likely due to either the complexity of the map creating inconsistencies, or falsely identifying matching landmarks when scanning the environment.

Conclusion

The SLAM algorithm used to solve the problems above has proven to be a reliable and powerful algorithm that allows one or more AVs to confidently navigate an unfamiliar terrain after some amount of exploration is conducted. For most of the data sets, only two iterations were needed to find a good estimate of the environment and its location within, showing its effectiveness at quickly optimizing the problem.

Its application to the real world only continues to grow as technology continues to advance and new methods are discovered. SLAM is a tool that will greatly help society in unimaginable ways by allowing for quick, accurate models of environments that might be too dangerous or inconvenient for humans.

Appendix

Below are some of the plots the resulted from the other datasets run in this script.

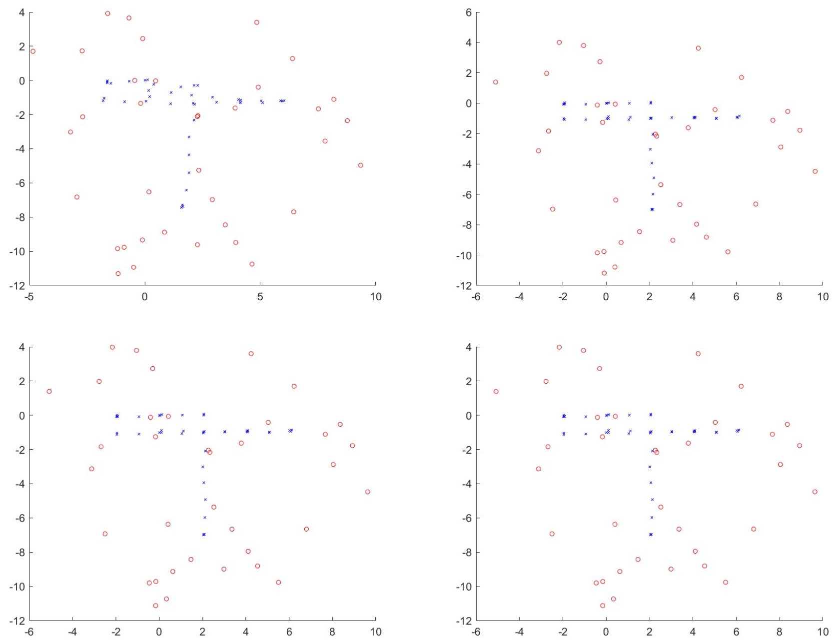

Figure A1 : plots of simulation-pose-landmark at iterations one, two, three and five (final) from top left to bottom right.

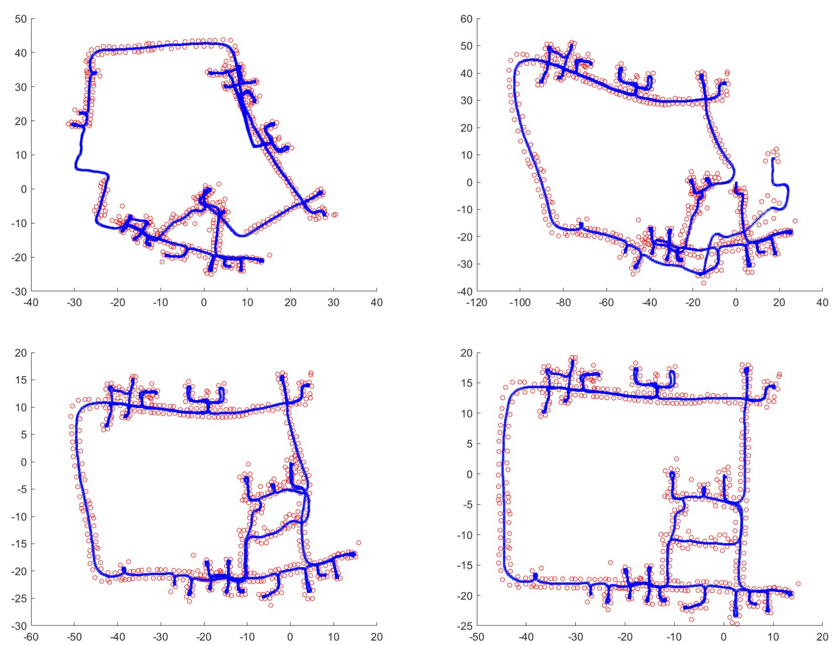

Figure A2 : plots of dlr at iterations one, three, five and eleven (final) from top left to bottom right.