Alignment of Centimeter Scale Bathymetry using Six Degrees of Freedom

Ethan Slattery, University of California Santa Cruz

Mentors: David Caress

Summer 2018

Keywords: point-clouds, iterative closest point, bathymetry, ROV, mapping

ABSTRACT

By using an ROV based, low-altitude mapping system equipped with LiDAR, multibeam sonar, and a stereo photography system enables the collection of high-resolution data in complex terrain. The vehicle developed at the Monterey Bay Aquarium Research Institute operates at a 3-meter altitude above the seafloor while traveling at speeds of 1 meter/second or less. The LiDAR system scans a 90° field of view 40 times per second, each scan containing 1600 individual soundings with an 8mm footprint on the seafloor. The resulting 64,000 soundings per second produces 1cm resolution bathymetr. This high resolution means that even small errors in data alignment result in significant mapping artifacts. Past techniques for alignment did not account for angular error, and therefore are insufficient for the alignment of this high-resolution data. This paper explores the use of Iterative Closest Point as an algorithm to align the native 3-dimensional data using both translation and rotation – six degrees of freedom.

INTRODUCTION

Low Altitude System Hardware

The Monterey Bay Aquarium Research Institute (MBARI) has developed a low altitude mapping system that consists of LiDAR, multibeam sonar, and a stereo photography system.

The LiDAR is a new system developed by 3D at Depth called the Wide Swath Subsea LiDAR (WiSSL). It consists of two laser heads each scanning a 50° field of view, overlapping by 5°. This results in a 90° field of view for the LiDAR system. Each laser footprint is 8mm and, achieves 1cm resolution when the vehicle is flying at 3 meters at less than 1 meter/second. The Sonar system is a 400-kHz Reson 7125 multibeam sonar that collects a 135° swath with 5cm resolution. The stereo photography system consists of two GX1920 2.4 MPixel color cameras that provide an 80° field of view, illuminated by dual strobes.

The estimated pose of the vehicle is provided by a Kearfott SeaDevil Inertial Navigation System (INS) and a 300kHz Teledyne RD Instruments Doppler Velocity Log (DVL). The offsets and rotations between the INS and the other sensors provide locations of each sensor allowing the projection of collected data into 3D space.

Figure 1 shows an example of the data products produced by this system. The products shown are a 50m square survey of sponge ridge in Monterey Bay.

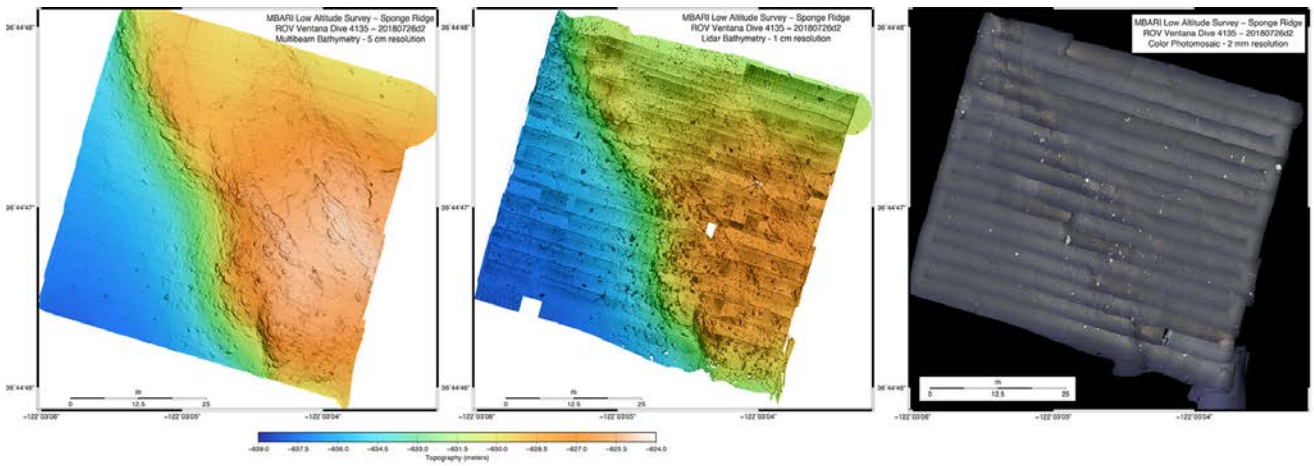

Figure 1: Low Altitude Mapping Products – multibeam sonar, LiDAR, Photomosaic

Low Altitude Survey Data Representation

Data collected from the LiDAR and sonar are represented in their native form as some combination of sensor location (offset from INS pose), time of flight to the bottom, and the angle of the specific beam or laser. The implementation of this for multibeam sonar very different from the LiDAR and the underlying implementation of both are beyond the scope of this paper. That raw data is used to create an unordered collection of points in space that represent the seal floor. Each point has a latitude, longitude, and depth to represent the location in space. Additionally, each point has a flag that indicates if it is a valid point. Invalid points can represent non-picked returns from sonar, noise, material in the water column, or other undesirable data.

This point data for a given survey is divided into sections, each section containing some amount of points along a single ROV navigation path that is easy to deal with algorithmically. These sections overlap each other and must be aligned to account for uncertainty in the INS pose and error in the sensor to INS offsets.

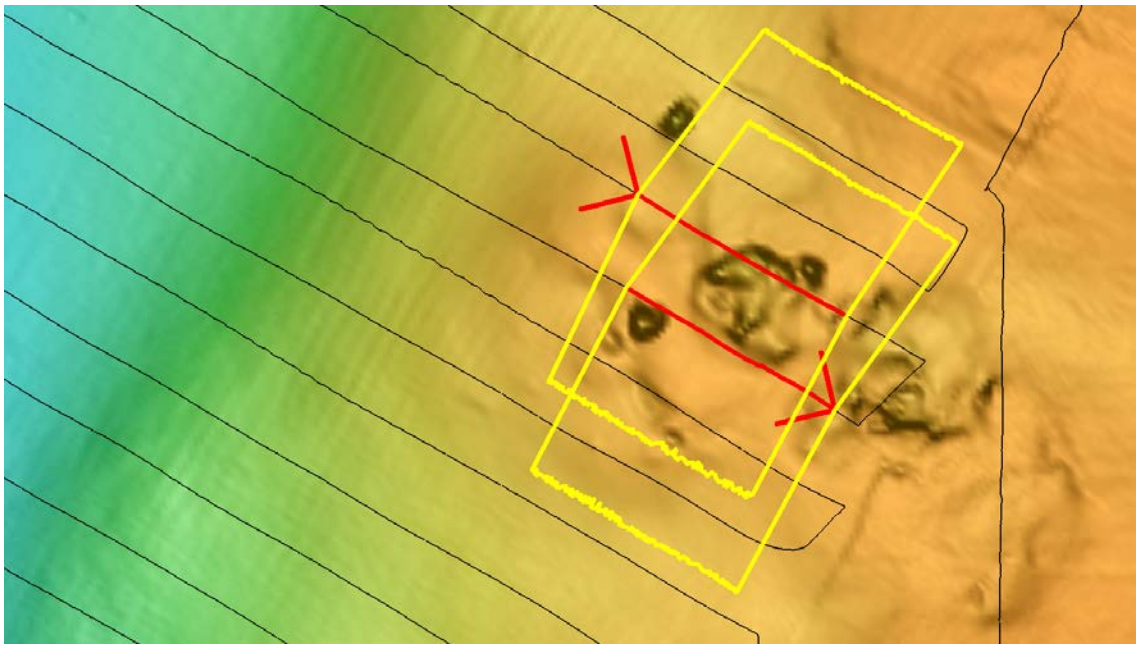

Figure 2: Example of two overlapping sections

Current Method of Data Alignment

Traditionally the alignment between sections was done using a gridded cross-correlation method. In this technique, each section is gridded, and each grid subdivision receives an X, Y, and Z value that is representative of all the points contained in that grid. The gridded representation of each section in a crossing are then superimposed on each other, and the difference between these representative values calculated in the X, Y, and Z direction, the difference between these values for various amounts of translation in each direction along the three axes. This gridding technique takes the native 3D bathymetry data and forces it into a 2D representation instead of working with the raw 3D data. Furthermore, it is only able to consider alignments using translation ignoring any potential angular error in the data.

Given the lack of angular adjustments in the gridded cross-correlation method, small errors in alignment have always been present in the data. These errors were not large enough to cause any issues with the 5cm scale multibeam sonar data, but in the 1cm LiDAR data misalignment of even a few millimeters can cause artifacts in the data. In Figure 3 the artifacts caused by data misalignment can be seen in the LiDAR data (on the right). The dark smudges parallel to the survey

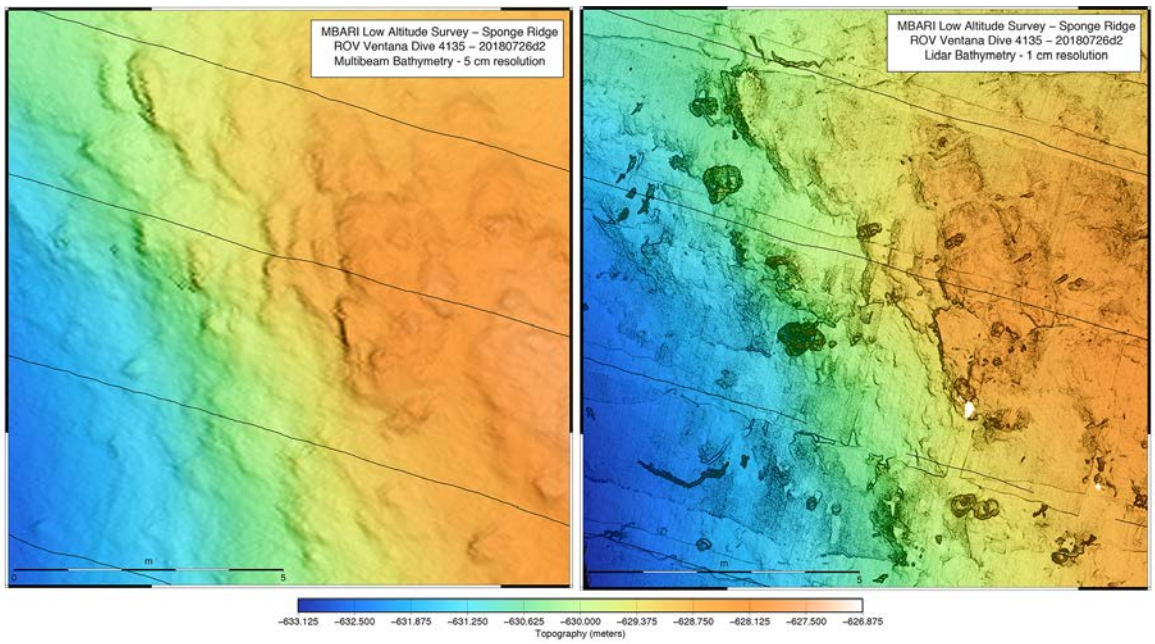

Figure 3: LiDAR misalignment artifacts compared to multibeam sonar

path, and the tearing perpendicular to the path are caused by the poor alignment. It is important to note that the same errors exist in the sonar data (on the left) but cannot be seen due to the lower data resolution.

Ideally, the alignment of data would be performed with an algorithm that uses the real 3D data and considers not only linear translation but also angular rotation when finding a best fit solution.

MATERIALS AND METHODS

The goal of this project was to align two overlapping sections of survey data. Each pair of sections with an overlap can then be aligned, and the transformation to get a good alignment saved. An overall solution can then be found using all the transformations from a given mission.

In each pair of sections there is a section that will stay static, and a section that will move to align with the static section. The static section is the *target* section, and the section that moves is the *source* section.

3D Alignment Algorithms

There are many algorithms for aligning point clouds, and much work has been done in the field of robotics with the aim of producing point clouds for use in Simultaneous Localization And Mapping (SLAM)[1]. The most popular and efficient is called the Iterative Closest Point (ICP) algorithm [2]. Given a rough alignment of the source and target clouds, ICP performs fine alignment by iteratively transforming the source cloud to minimize the sum of the squared distance between corresponding points in the two clouds.

There are many versions of ICP with varying effectiveness and efficiency on various types of point cloud data[3], but the most basic point-to-point ICP algorithm was chosen for the bathymetry data. The point-to-point method was chosen because it required the least amount of data translation, and its generic nature leads to good results without extensive testing.

3D Data Representation

Both the sonar and the LiDAR produce data that is a collection of points in 3D space. Normally this data is called a point cloud, and the first challenge was to get the MB-System data into a point cloud representation. Implementing a translation between the native MB-System format and a more standard point cloud format allowed the use of a wide array of open-source algorithms for working on point clouds.

Seafloor survey data is collected from a moving platform with positional data completely reliant on the INS. The raw soundings need to be transformed from a measure of angle and distance into a latitude, longitude, and depth based on the pose of the survey vehicle and the sensor offsets. Fortunately, MB-System already contained the code to perform these calculations. The existing code was used to take the raw survey data and create a vector of points with coordinates in latitude, longitude, and meters of depth.

Next, the point clouds were created from these vectors but in a local cartesian reference frame. This reference frame was created using the software library PROJ4 [4], transforming the lat-long coordinates for each point into a Cartesian coordinate system with the origin located on the center sensor location of the source section.

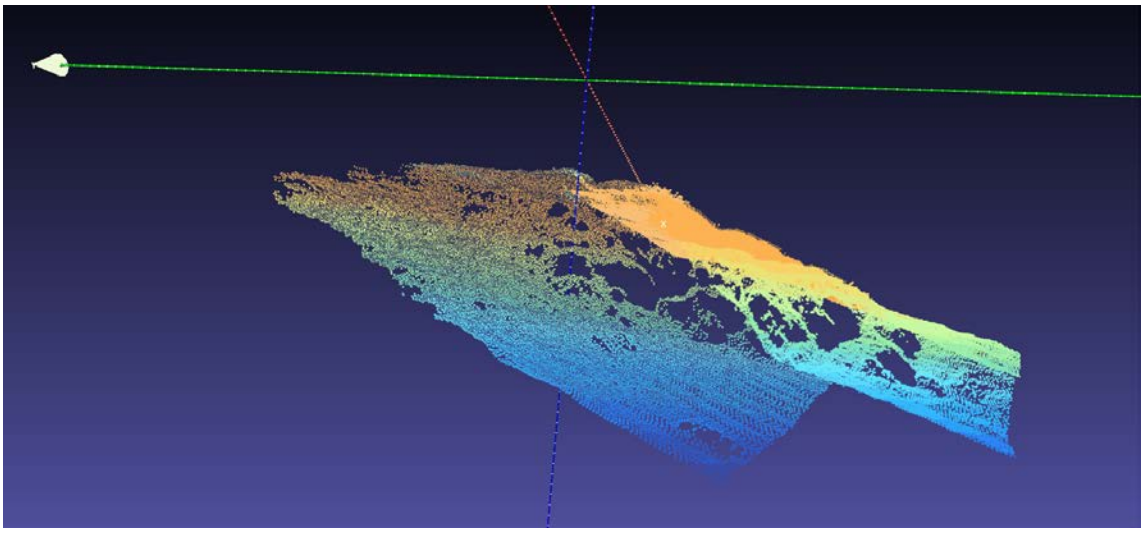

Figure 4: Source and Target points clouds in a local cartesian reference frame

Rough Alignment

Once the data is loaded into source and target point clouds, with coordinates in a local Cartesian reference frame, the first step is to get a rough alignment. Since the ICP algorithm finds the local minima of fitness, the two sections must be "close" to start. The definition of close depends on the data, but generally, the alignment must be closer to the true solution than any possible false solution. There are many ways to do this using the point cloud data, but since we already have a translation alignment from the gridded cross-correlation method associated with every crossing, that was used instead. MB-System stores the translations as a movement in meters along the X, Y, and Z axis. Since these axes are preserved between the world reference frame and the local frame, these translations are combined into a homogeneous transformation matrix (4x4 affine matrix) and applied to the source section.

Fine Alignment

Once the rough alignment is applied to the source the translation matrix used is retained, and ICP is started. Several implementations of the basic point-to-point ICP were tested, and the Point Cloud Library (PCL) [5] implementation was chosen. PCL is an open source library that provides a suite of tools for working with 3D point data. Not only does it provide an ICP implementation, but it also provides a convenient point cloud class, search algorithms for finding points, geometric algorithms, and simple visualization routines.

Since the PCL is open source and written in C++, the ICP implementation was inherited as the base of a custom alignment class. This allowed for the reimplementation of specific parts of the algorithm. The method of measuring fitness between the source and target sections was re-implemented, and several other methods were added to ease the interoperability between PCL and MB-System.

By default, the PCL implementation of ICP measures its fitness by using the sum of the square distances between every point in the target section and the nearest point in the source section. Since every crossing overlaps by a different amount,

this leads to a large amount of variation in fitness numbers that can not be compared between different crossings. A crossing with a large percentage overlap would have a very small fitness value, while a crossing with a small amount of overlap would have a large fitness value despite both being aligned correctly. To fit this issue, the distance was only measured between points that PCL had tagged as correspondence points, points that match between sections. This means that only points in the overlapping region are considered when calculating fitness. This gives a fitness value that can be compared between different datasets. These correspondence points are chosen by PCL initially based on distance; further filtering of correspondences is done by distance and enforcing a one-to-one relationship.

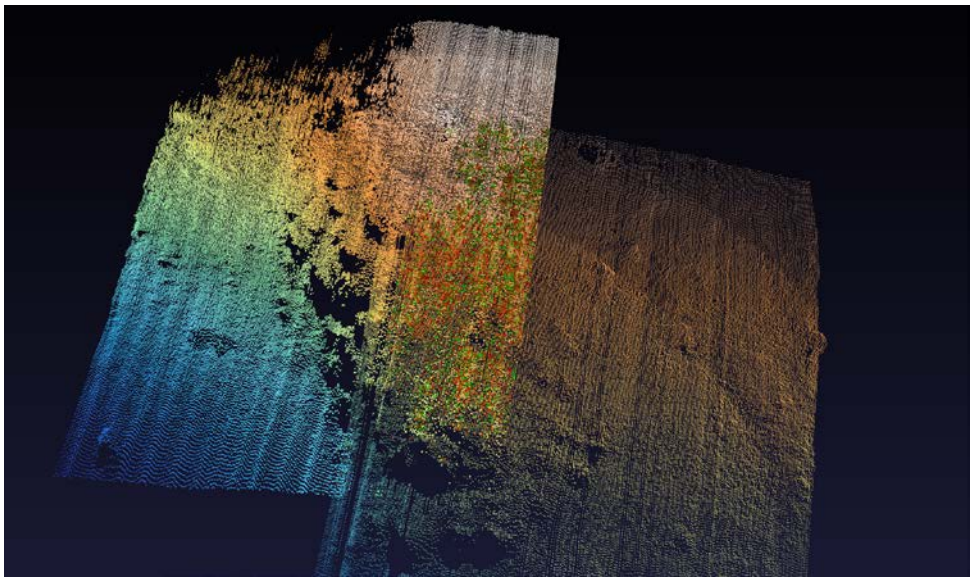

Figure 5: Aligned sections with correspondences highlighted in green and red in the overlapping region.

The PCL is then allowed to iterate the ICP algorithm a certain number of times, or until the fitness falls below a certain threshold, or the difference between two successive transformations falls below a threshold. These three parameters are configurable on a per-project basis and by default are 50, $0.0001m^2$, and 0.001m respectively. Once any of these conditions are met the final transformation matrix is obtained by getting the ICP transformation matrix from PCL and multiplying it with the rough alignment matrix.

The program then outputs the results as a single comma-separated line with the matrix, the transformation as both translation and Euler angle rotations, and some information such as some correspondence points used, and time spent in ICP.

RESULTS

This MB-System version of ICP using PCL was used to process two surveys. Both surveys consisted of several hundred crossings and were processed using eight threads on an eight-core x86 processor, with each crossing taking from 1 to 120 seconds to process. The results were then analyzed using the R statistical software package. The distribution of fitness results before and after running ICP was plotted as a density plot. A density plot was also created to show the distribution of translation and rotation magnitudes required for alignment. Recall that fitness is the sum of the squared distance between corresponding points in the target and source sections

The first was a survey in Santa Monica conducted from the R/V Rachel Carson using ROV Ventana on the 19th of May 2018. It consisted of 931 crossings, and there were known issues with alignment due to some hardware issues on the mapping sled. Using the gridded cross-correlation method only, the fitness of the crossings had a mean score of 0.025m^2 with a standard deviation of 0.1. After running ICP, the mean score was 0.0013m^2 with a standard deviation of 0.0019.

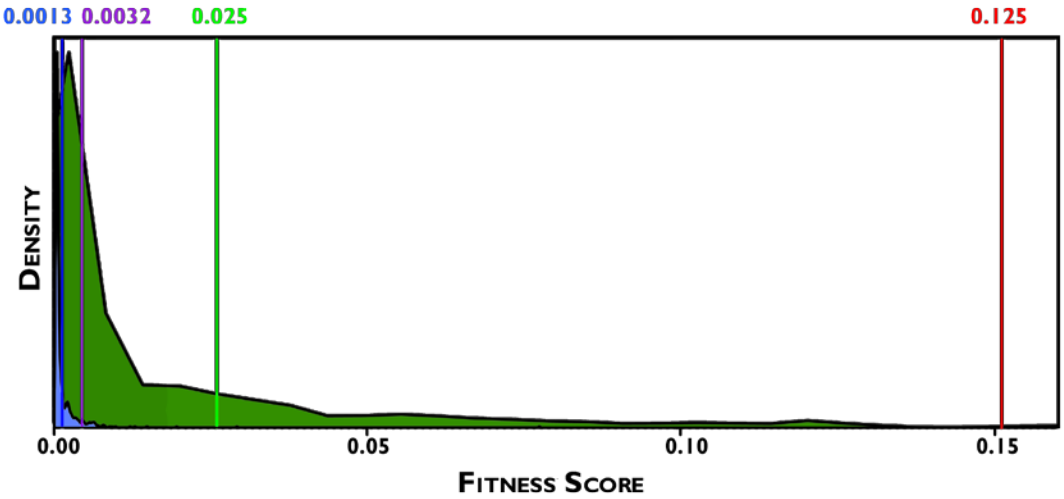

Figure 6: Fitness distribution of the Santa Monica survey before (green) and after (blue) performing ICP algorithm

The second survey was conducted in Monterey Bay on an underwater feature named Sponge Ridge. The survey was conducted from the R/V Rachel Carson using ROV Ventana on the 26th of June 2018. This survey was smaller with only 97 crossings, but the hardware problems in the system were corrected resulting in much better rough alignments. Using rough alignments, the mean fitness value was 0.00056m^2 with a standard deviation of 0.00134. After running ICP, the mean score was 0.00027m^2 with a standard deviation of 0.00029.

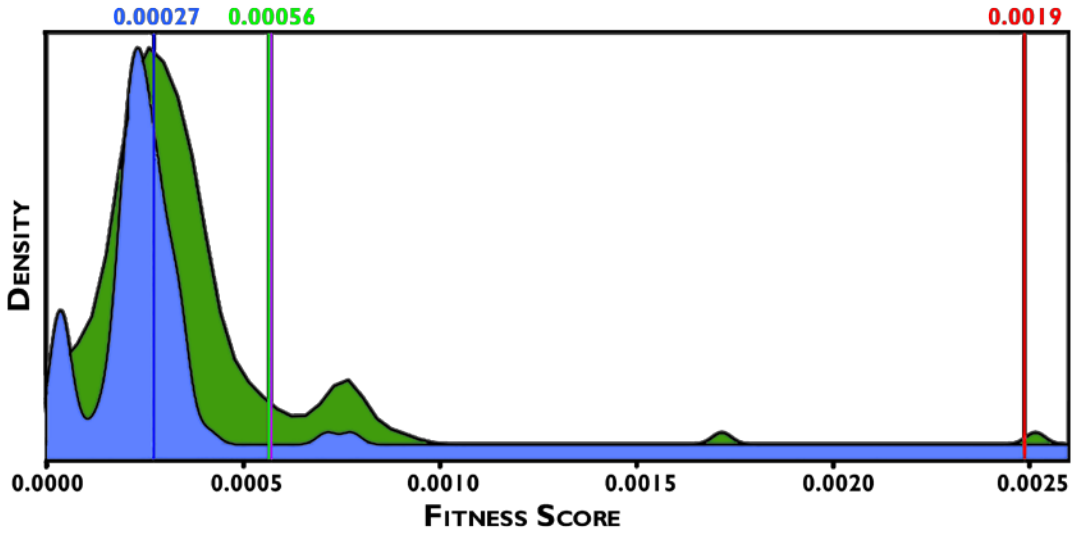

Figure 7: Fitness distribution of the Sponge Ridge survey before (green) and after (blue) performing ICP algorithm

It is also interesting to look at the distribution of adjustments made to obtain the correct alignment. The distribution of translations and rotations for the 97 crossings in the sponge ridge survey can be seen in Figure 8. The mean is marked in green and the first standard deviation marked in red on each plot. As expected the majority of both the translation and rotation is on and around the X and Y axis (latitude and longitude). This is expected since INS drift will affect those directions most. Error in Z translation is dominated by the depth meter, and rotational error about the Z-axis is dominated by the compass heading. Both these sensors should be much more accurate over time than the other INS measurements.

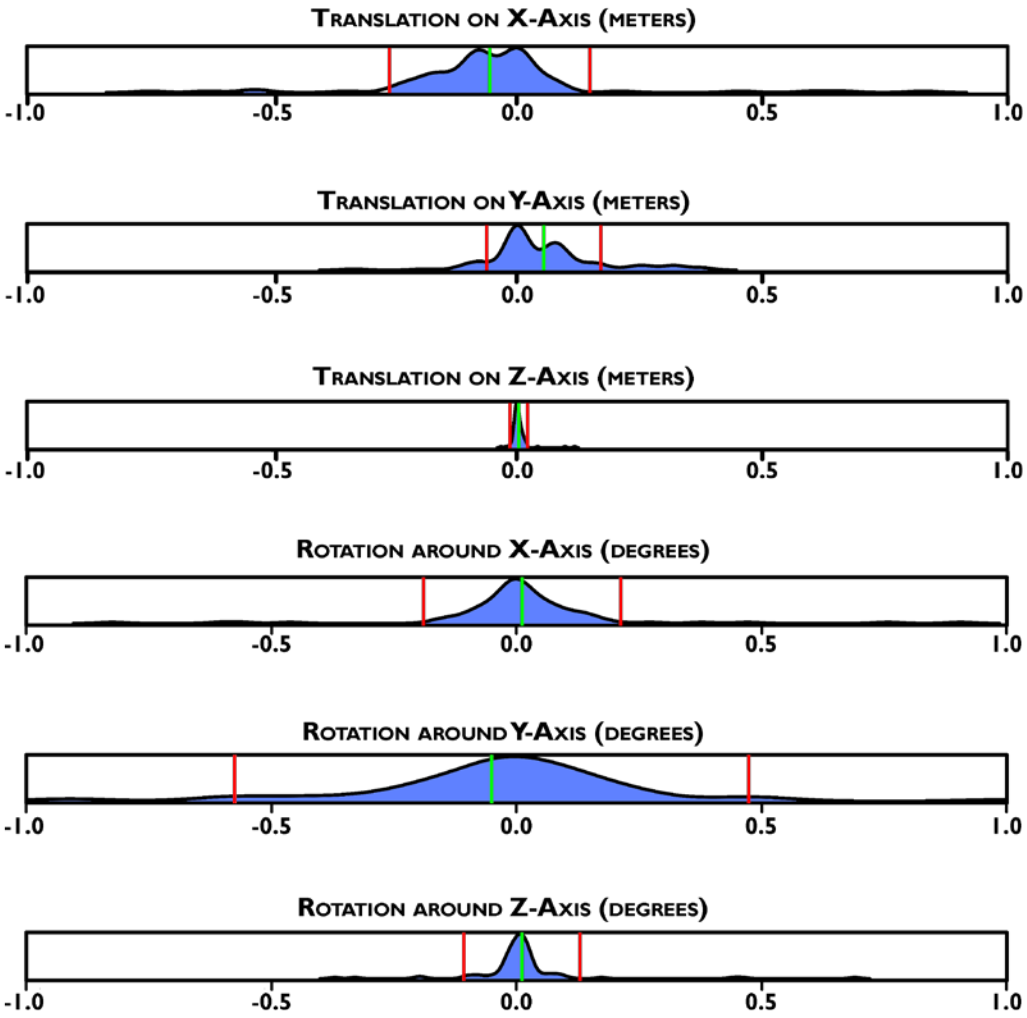

Figure 8: Distribution of translation and rotation correction for crossing in the Sponge Ridge survey

DISCUSSION AND FUTURE WORK

MB-System is an open source and comprehensive ocean-floor mapping software suite that is maintained at MBARI. The eventual goal is to fully integrate the ICP algorithm into the MB-NavAdjust module that is responsible for aligning data and solving for a navigation solution. As the code currently exists, it can be called to perform ICP on a single crossing or to process an entire project, and the output on standard-out can be parsed for the alignment solution. Eventually, the code should be used as a library and called from within MB-NavAdjust and distributed with future versions of MB-System.

The alignments created using ICP are currently performed in the local reference frame, and the adjustments are described as translations and rotations about the local X, Y, and Z axes. These are roughly aligned with latitude and longitude, but *not* aligned with the axes of the sensors. Ideally, transformations would be produced using the sensor as the origin, with X/Y/Z rotations representing Roll, Pitch, and Yaw. Knowing the transformation from a frame where the origin is on the sensor in the source and also in the target section would lend to the calculation of sensor offsets and an overall navigation solution.

CONCLUSIONS/RECOMMENDATIONS

In the two surveys processed the mean fitness was cut roughly in half, and the standard deviation substantially reduced. This shows that the alignment of bathymetry data collected with the low-altitude mapping system can be improved by using point-cloud based algorithms which consider translation and rotation. It also serves as a starting point to integrate the PCL into the larger MB-System to allow direct analysis of the 3D data with six degrees of freedom instead of forcing the data into a gridded representation. With some additional work, the alignments obtained from ICP alignments could be used to solve for sensor offsets and a better navigation solution.

ACKNOWLEDGEMENTS

Special thanks to Dave Caress for his mentorship and accommodation during the summer internship 2018, and to George Matsumoto and Linda Kuhnz for organizing the summer internship program. This research was made possible through their hard work along with the David and Lucile Packard Foundation, whose funding makes the work MBARI does possible. The crew of the R/V Rachel Carson also deserves a special thanks for accommodating me as a summer intern a total of six times over the summer, the work they do is amazing.

References:

[1] J. Aulinas, Y. Petillot, J. Salvi, and X. Lladó, “The SLAM problem: A survey,” in *Frontiers in Artificial Intelligence and Applications*, 2008.

[2] F. Pomerleau, F. Colas, and R. Siegwart, “A Review of Point Cloud Registration Algorithms for Mobile Robotics,” *Found. Trends Robot.*, vol. 4, no. 1, pp. 1–104, 2015.

[3] Benbellekens, Vincentspruyt, Rafaelberkvens, and Rudipenne, “A Benchmark Survey of Rigid 3D Point Cloud Registration Algorithms,” *InternationalJournalonAdvancesinIntelligentSystems*, vol. 8, no. 1, pp. 118–127, 2015.

[4] PROJ contributors, “PROJ coordinate transformation software library.” 2018.

[5] R. B. Rusu and S. Cousins, “3D is here: Point Cloud Library (PCL),” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2011.