Photomosaic Mapmaking of the Seafloor using SIFT

Marcus Hammond
Aerospace Robotics Laboratory
Stanford University
marcus.hammond@stanford.edu

Abstract

This project addressed the problem of creating mosaic maps of the seafloor using remotely-operated vehicles (ROV) or autonomous underwater vehicles (AUV) operated by the Monterey Bay Aquarium Research Institute (MBARI). Due to the attenuation of light under water, maps of large areas must be created by stitching together close-up images of the sea floor into a self-consistent, spatially accurate map. In this project, the use of SIFT features to perform image matching and registration was investigated. Bundle adjustment was performed to estimate 6-DOF affine transformations to place each image in the global frame. Additionally, several concepts for a real-time tool for helping ROV pilots ensure that they are achieving complete coverage were explored.

1. Introduction

1.1. Photomosaic maps

Maps of the sea floor are used widely for everything from studying coral reef ecology to underwater archeology and shipwreck documentation.[7] Additionally, these mosaics can be used as a navigation aid, even when the map itself may not be the end goal. GPS signals cannot penetrate the ocean to provide localization information. Long-baseline acoustic localizer arrays can provide accurate position information, but are expensive, both monetarily and logistically.[5] Vision can be used to augment inertial-based dead-reckoning navigation. However, like dead-reckoning, vision-based navigation is prone to drift without bound. However, with vision this drift can be handled by flying self-intersecting trajectories, essentially knocking out accumulated error by matching image features to images taken before the error had a chance to grow. A good discussion of this is provided in [2]

1.2. Current Approach

MBARI currently performs mosaic mapping of the seafloor by running a remotely operated vehicle in a back-and-forth "lawnmower" path, grabbing frames from a video camera as it goes. The resulting images are correlated using Signum Laplacian of Gaussian (SLoG) filtering as described by Richmond [7]. This method is not robust to scale or orientation, and can only be done robustly with the high frame rate provided by video input. This is acceptable for performing registration in-line, but performs weakly when attempting to perform side-to-side correlation between two swaths. The current system requires offline verification of swath overlap *after each swath*, which is slow, and not robust to scale, orientation and illumination changes.

1.3. SIFT image features

SIFT features, developed by David Lowe [4] are the state of the art when it comes to detecting and describing stable image features. They are robust to large changes in scale and orientation. Also, because each feature has a unique 128-dimension identifier, correlation can be performed between photos taken far apart, both in time and space. These are two instances in which SLoG correlation performs poorly. Once feature correlations are established, robust algorithms that reject outliers such as RANSAC or Hough transform can be used to register the images in the map.

2. Data and Validation

2.1. Experimental Data

For this paper, MBARI provided dive data from December 8, 2009. During this dive, an ROV was driven in a "lawnmower" path 30 meter squared area at a height of roughly 2 meters. Navigation metadata is also provided, giving the ROV's best estimate of its own pose and velocity at the time each picture is taken, as based on data from the inertial measurement unit (IMU) and doppler velocity logger (DVL). These sensors are quite accurate, with drift rates

on the order of 1% of distance traveled or less. However, the presence of any drift at all makes vision based environment-relative position measurements desirable.

2.2. Validation

For the reasons stated above, no absolute positioning data is available. To validate the success of the map, two metrics will be considered: image feature reprojection error and qualitative self-consistency as evaluated by human observer.

With a perfect model, all matching points observed in separate image frames would map to the same location in the global reference frame. The mean squared-error of this reprojection is a good metric of model quality.

Scattered throughout the mosaic area are floats anchored to the ground with heavy weights. While these floats may not generate consistent and robust image features, and bob and sway with the currents, making them useless for image registration, they do provide a good reference for human evaluation of the map. Because the anchor point of the float is fixed to the seafloor, each separate image of a given float should appear to be anchored at the same point in the final mosaic. If the different images of the floats do not appear to share a common root point, this is symptomatic of model error.

3. Approach

3.1. SIFT-based correlation and registration

In this paper, SIFT features were used to perform correlation between images. SIFTs were detected in each image using the open-source toolbox vlfeat [8] If an empirically-determined minimum number of features were observed in an image (good visual lock), the vlfeat implementation of Lowe's feature matching function as described in [4] was used to match the image frame to other nearby frames. The matches were then fed to a function that uses a Random Sample Consensus (RANSAC) algorithm to estimate a pairwise transformation between the images. If a model can be determined successfully, the inliers are assumed to be good matches and will later be used to estimate global transformation parameters.

3.2. RANSAC

3.2.1 Implementation

The RANSAC code used in this study to determine whether point matches are valid was a modified version of code provided to the author by David Chen, a graduate student in the Computer Science department at Stanford. Mr. Chen worked as a course assistant for EE 368, and image processing course, and provided the code in that context. For this study, the code has been modified to find a variety of different models for transforming images into different reference frames. To generate a model that is robust to outliers, the RANSAC algorithm selects a subset of matches and solves for the transformation parameters that best map the features in one image into the feature locations in the other frames. If the model is linear, this is done using least-squares. If it is not linear, an iterative process must be applied. In this paper, nonlinear least-squares was used to find these iterative solutions. The robustness of the RANSAC algorithm can be seen in figures 1 and 2. In figure 1, there are "true" matches and "false" matches, most notably at the top and bottom of the images. After applying RANSAC to estimate an 8-DOF perspective transform, these bad matches have been rejected, as shown in 2.

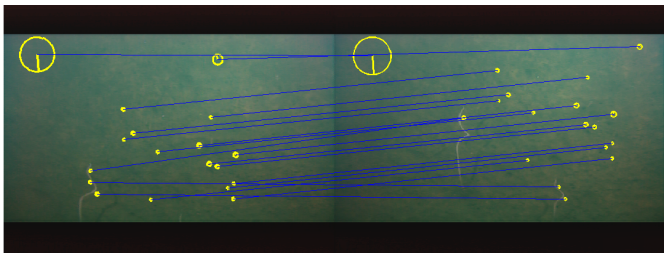

Figure 1. Though several false matches occurred between images, the majority of the SIFT matches show consistent frame-to-frame correlation

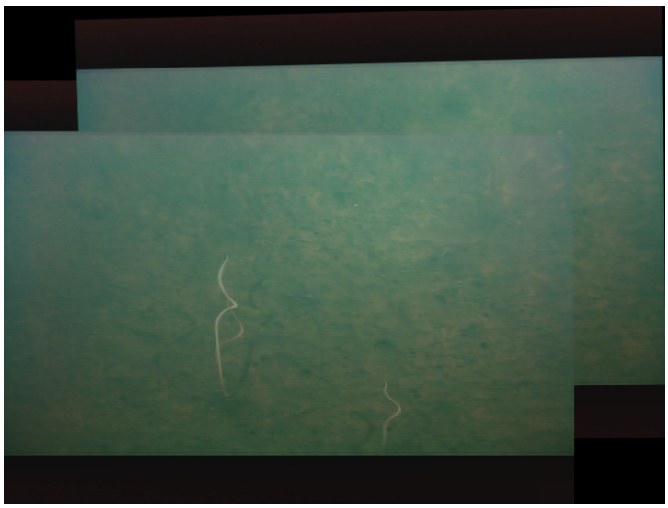

Figure 2. The RANSAC algorithm rejected the false positive matches, yielding a good transformation model between the two images in figure 1.

3.2.2 Shortcomings

If there are few matches, RANSAC will not to a good job at coming up with a model that describes the transformation. Furthermore, if the ratio of false matches to true matches

gets larger than roughly 1:1, RANSAC again will struggle. A different approach for determining whether good overlap occurs between frames could be a Hough-like approach, as put forth in [4]. For further discussion of this, please refer to the "Future Work" section.

3.3. Model Options

Given a model H^{ij} for transforming image coordinates between images i and j, the following equation should hold.

$$\begin{bmatrix} x^j \\ y^j \\ 1 \end{bmatrix} = H^{ij} \begin{bmatrix} x_1^i \\ y_1^i \\ 1 \end{bmatrix} \tag{1}$$

If we have an H^{ij} matrix of the following (affine) form

$$H^{ij} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

,

this problem can be rearranged into the form

$$\begin{bmatrix} x_1^j \\ y_1^j \\ \vdots \\ x_n^j \\ y_n^j \end{bmatrix} = \begin{bmatrix} x_1^j & y_1^j & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1^j & y_1^j & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^j & y_n^j & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_n^j & y_n^j & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} \tag{3}$$

Where n is the number of matching points in images i and j. This is a least squares problem that can then be solved to find the parameters of the transformation matrix.

A number of different models to register the frames were investigated. Each has advantages and disadvantages. They are each described here, in order of increasing complexity.

3.3.1 Translation only

In this model, only two parameters are estimated: x- and y-offset.

$$H_t = \begin{bmatrix} 1 & 0 & x_o \\ 0 & 1 & y_o \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

This is the simplest and fastest model. If the robot is moving in a lawnmower path while holding a constant heading and altitude, this method can be surprisingly successful. If the robot does not maintain constant heading or altitude, but a reliable estimate of both is known, the image can be predistorted before estimating the two parameters. However, if there are large unmodeled in-plane rotations or elevation changes, this model will fail miserably.

3.3.2 Translation and scaling

In this model, three parameters are estimated: two translation parameters and a third for global scale change.

$$H_{ts} = \begin{bmatrix} s & 0 & x_o \\ 0 & s & y_o \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

This model can handle unmodeled elevation change, which causes apparent scale change, but still fails under unmodeled rotation.

3.3.3 Translation and orientation

This model still only estimates three parameters, x_o, y_o and θ. However, the constraint that the rotation matrix be orthogonal is nonlinear, and a simple least-squares approach to solving for the parameters cannot be used. Instead, a nonlinear least-squares approach was used with good effect. Starting with an estimate of the nominal $\theta = 0$, the solution generally converged within ten iterations. Further discussion of this method can be seen in [1]

$$H_{t\theta} = \begin{bmatrix} \cos\theta & -\sin\theta & x_o \\ \sin\theta & \cos\theta & y_o \\ 0 & 0 & 1 \end{bmatrix} \tag{6}$$

3.3.4 Translation, orientation, and scale

This model combines the previous two, though now the orthogonality constraint is no longer in place, allowing a linear solution once again.

$$H_{\text{similarity}} = \begin{bmatrix} a & -b & x_o \\ b & a & y_o \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

3.3.5 Affine

An affine model is the highest accuracy of the pseudo-linear models. Since the 3rd (homogeneous) term is guaranteed to be one, no dividing by the third term is necessary. Hence, affine models can still be estimated with linear least-squares methods. This simplifies the math, while still approximating the accuracy of higher-fidelity perspective models for small out-of-plane rotations.

$$H_a = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \tag{8}$$

3.3.6 Higher-fidelity models

Using an 8-DOF perspective model assumes, as all the previously mentioned models, that all features lie in a plane, but allows for large out-of plane rotations. However, the nonlinear division by the scale factor, w makes the math much more complicated.

$$H_p = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \tag{9}$$

$$\begin{bmatrix} wx^j \\ wy^j \\ w \end{bmatrix} = H_p^{ij} \begin{bmatrix} x_1^i \\ y_1^i \\ 1 \end{bmatrix} \tag{10}$$

At the time of writing, camera calibration to correct for any nonlinear distortion parameters due to lens effects had not been attempted. Augmenting the parameter estimation to determine these terms could yield a more accurate reconstruction, though at the expense of more sophisticated optimization techniques.

Even more complicated models, such as structure from motion [3] model each image feature as existing in 3-D space, not constrained to a plane. However, these techniques generally require features to be seen very many times, with some relying on the assumption that all features can be seen in every image. This assumption is not valid when translating a camera over large distances while close to the object being imaged, as with photomosaicking. However, the mapmaking process may benefit from some of the ideas set forth in SFM. Please see future work for further discussion.

4. Results

4.1. Image correlation density improvements

As shown in figure 3, SLoG only produces a few side-to-side correspondences (red links). Furthermore, these side-to-side links are expensive to calculate and require a good starting estimate of translation in order to select subregions of images with which to perform the 2-D correlation. Using SIFT matching, not only can many more cross-links be detected, but it can be done nearly real-time. This greater degree of interconnectedness between frames allows for better accuracy when performing bundle adjustment to estimate global transformation parameters. The fast speed with which these links can be detected allows ROV pilots to monitor the amount of overlap their paths are achieving, so that they can achieve full coverage in the final map.

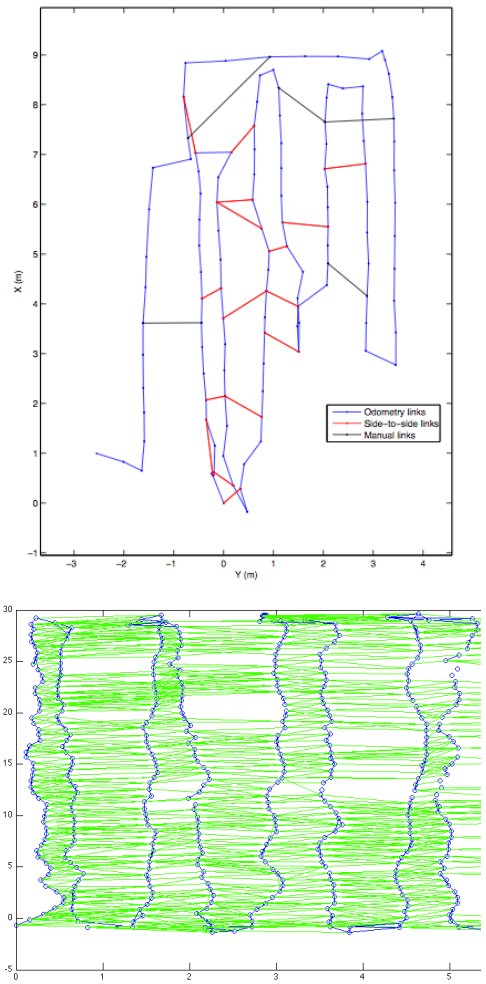

Figure 3. Side-to-side links using SLoG (top, Richmond [7]) vs. SIFT. SIFT has a much higher density of detections between frames.

4.2. Model selection for online parameter estimation

Choosing a camera model is an important step in building a mosaic. Thus far, five different image transformation models have been tried: translation only, translation with in-plane rotation, translation with scaling, similarity transform, and affine.

While performing batch processing to minimize global reprojection error is an option for offline map optimization, it is also desirable to produce a map of reasonable accuracy in real-time as an ROV pilot aid while acquiring data. This consideration pushed the use of the above models. Other methods for image stitching similar to the one proposed use full perspective camera models, most notably the autostitch algorithm used by Brown and Lowe [1] However, his method assumes that the camera is not translated, only rotated about its optical center, making it possible to relate the

transformations between successive frames through simple matrix multiplication. The large translations involved in mosaic mapmaking violate this assumption, and necessitate another approach. Using the transformations mentioned above, the global frame transformations can be calculated simply by multiplying their between-frame transformation matrices.

Figure 4. The most successful model for online estimation was the simplest, where only two parameters, x and y translation are estimated

The results are somewhat surprising. The most successful method for online map building is the one in which only the x and y translation parameters are estimated. This can be seen in figure 4. The author believes there is unmodeled pitch and possibly yaw bias in the camera that causes apparent looming and strafing motion in the field of view that is not uniform throughout the image. Adding degrees of freedom to the model allow it to try to account for this, but since the error is not zero-mean, it compounds.

Possible solutions for this are to try to estimate these parameters and predistort each image before extracting SIFT features and performing correlation, in effect "knocking out" the error so it cannot propagate. If measurements of orientation and altitude are available, which is generally the case, these parameters can be leveraged to extend the 2-DOF estimation to more general trajectories.

The only reason the 2-DOF model works at all is that the ROV is flying a constant-heading–constant-altitude path. If either of these were not the case, the author expects that the 2-DOF model would fail.

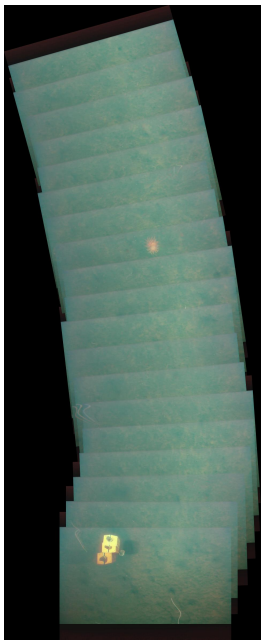

Figure 5. When in-plane rotation ψ is also estimated, the swath begins to drift. The author suspects that a pan-tilt camera bias is causing this behavior.

Figure 6. The affine model also falls victim to the distortion seen in the rotation model, though its manifestation is different. Each successive frame suffers from a greater degree of skew, rotation, and scaling.

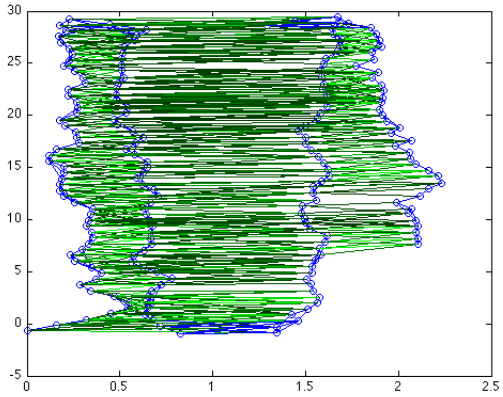

Figure 7. This is an abstract tool for visualizing overlap. Green lines are drawn between nodes on a plot of the vehicle path, showing where enough matches to create a model occur. The shade of green corrseponds to the number of matches, with darker indicating more matching features. This is intended for use by ROV pilots to show them in real time how well they are achieving coverage.

Figure 8. This map was generated in real time using a 2-parameter model. Areas of overlap were highlighted by boosting the red values of the pixels by a small amount. The effect is additive, so areas of multiple frame overlap are more red. This is intended for use by ROV pilots to show them in real time how well they are achieving coverage.

4.3. Visualization tool for real-time pilot aid

One goal of this project is to give ROV pilots a tool for gauging how well they are achieving coverage of a given section of sea floor in real-time. Figure 8 shows one conception of such a tool. This tool will allow MBARI to create maps with more confidence and autonomy from the ARL.

Two concepts were explored for overlap visualization. The first, figure 7, is the more abstract of the two. The x-y positions of the robot along its path are plotted, and lines are drawn between frames where many feature matches are detected. The shade of the connecting line indicates how many matches are found.

The second concept takes a more intuitive visual approach, as seen in figure 8. Since fine accuracy is not of paramount importance for this navigation, frames are laid down in real-time using the 2-DOF model, and any areas that overlap with previous slides are highlighted in red. The effect is additive, so areas of multiple overlaps appear a brighter red.

As of the time of writing, these are both only concepts awaiting ROV pilot feedback to determine what information they prefer to use.

4.4. Global parameter estimation via bundle adjustment

4.4.1 Model selection for bundle adjustment

An affine model was selected for global parameter estimation. This allowed the model to capture more detail than the simple translation-only model, without complicating the math too much. Since the out-of-plane rotations were quite small, moving to a perspective model would have complicated the calculations for determining the parameters without gaining much in terms of reprojection error reduction. Please refer again to equation 3 for the basic building block of this optimization. To see how these building blocks combine to produce a least-squares type problem for the parameter estimation, consider the following small problem, where parameters for only two images are to be estimated. We will refer to these global parameters as a_i through g_i, where i is the image number. If we write equation 3 as

$$\begin{bmatrix} x_{ij}^g \\ y_{ij}^g \end{bmatrix} = A_{ij} p_i \tag{11}$$

$$A_{ji} p_j = A_{ij} p_i \tag{12}$$

$$\tag{13}$$

,

which can be written in block matrix form as

$$\begin{bmatrix} 0 & 0 \\ 0 & A_{ji} \end{bmatrix} \begin{bmatrix} p_i \\ p_j \end{bmatrix} = \begin{bmatrix} A_{ij} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} p_i \\ p_j \end{bmatrix} \tag{14}$$

The optimization then boils down to a problem of the form $0 = Ap$, where A is a large sparse matrix with an A_{ij} appearing any time images overlap and p is a tall vector consisting of all image transformation parameters.

In order to avoid the trivial solution where all images are mapped to the origin, this problem needs to be constrained. A simple but clever constraint proposed in [6] was used. This constraint penalizes changing the magnitude of the diagonals of the image, thereby limiting the amount of scaling and skewing the optimizer will permit, while leaving it free to perform large rotations and translations. Because the robot is flying a a roughly constant altitude and nominally pointed normal to the floor, the bias toward unit scale and small skew introduced by this constraint is of little consequence. In practice, this constraint is implemented by appending equation 15 to the equation $0 = Ap$ for each parameter set p_i.

$$\begin{bmatrix} x_{tl-br} \\ y_{t-br} \\ x_{tr-bl} \\ y_{tr-bl} \end{bmatrix} = \begin{bmatrix} x_{tl-br} & y_{tl-br} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_{tl-br} & y_{tl-br} & 0 \\ x_{tr-bl} & y_{tr-bl}) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_{tr-bl} & y_{tr-bl} & 0 \end{bmatrix} [p_i] \tag{15}$$

where x_{tr-bl} is the x component of the top-rightmost pixel minus the bottom-leftmost.

4.4.2 Validation of bundle adjustment and areas for improvement

Using bundle adjustment to estimate all image parameters in a batch process, we were able to produce an affine model that did not drift or skew excessively. Due to data storage limitations running Matlab on a laptop computer, it was only possible to for a mosaic for about half of the data available at a time. Any more and the program tended to crash. The reprojection mean squared error for the mosaic showin in figure 9was equal to 64 pixels^2. This means that on average the mapped features were misaligned by about 8 pixels. For reference, the overall size of this mosaic is 3030×5879 pixels. This is not particularly good, but consistent with the fact that the images are not in any way transformed to account for lens distortion or other intrinsic camera imperfections.

Using the more qualitative/subjective validation technique, the map appears to be more self-consistent in some regions than in others. For instance, floats near the periphery of the mosaic all tend to fan out from a common anchor point, whereas floats nearer the middle exhibit more misalignment, appearing to be "smeared" across the floor. The effect can be seen in both figures 9 and 12 At the time of writing, the cause of these effects remains unknown.

Overall, the self-consistency of the map ranged from around 2 inches at best to two feet at worst, once outliers were rejected.

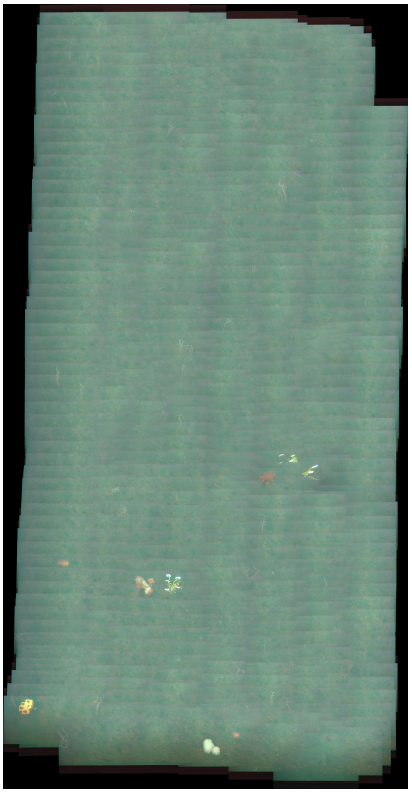

Figure 9. First 828 images in mosaic assembled using affine model found through bundle adjustment. The inconsistency in registration accuracy can be seen in the two clusters of floats.

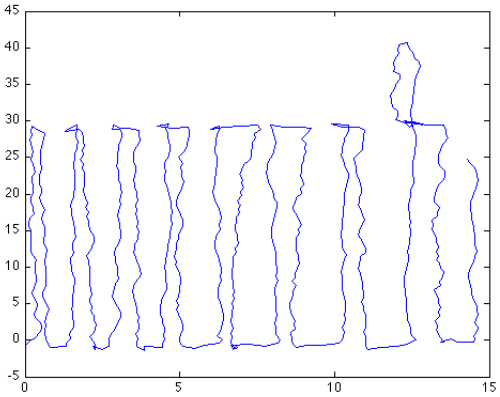

Figure 10. Ground track of the robot trajectory for the first 1000 images. Notice the loop near the top right where the pattern was broken. This area caused some problems for the optimizer.

At one point during the run, the robot makes a loop outside of the 30 m^2 survey area (see fig. 10), the start of which corresponded to an image frame with low feature content. There was still enough overlap between frames in the loop

to create a self-consistent sub-map, as shown in figure 11. However, since the loop did not have a good link with the rest of the map due to the loss of visual lock, the optimizer lost track of where to put this sub-map with relation to the rest of the frames. This raises a good question: "What should we do when visual lock is lost?" This will be discussed in the future work section.

Figure 11. Loop from errant trajectory

5. Future work

During the course of this project, a number of paths for future research and improvement were identified.

First and foremost among these problems is the issue of mis-regsitration of the images. Specifically, we need to identify the reason why some parts of the map appear to register better than others. The author believes that doing this will require estimating intrinsic camera nonlinearities, as well as improved fault detection and handling when visual lock is lost.

Building on the idea of fault detection and handling, the current system for identifying links between image frames could be improved upon greatly. Under the current framework, much prior information is discarded, leading to problems like the global misplacement of the errant loop submap discussed in figure 11. By developing methods to detect

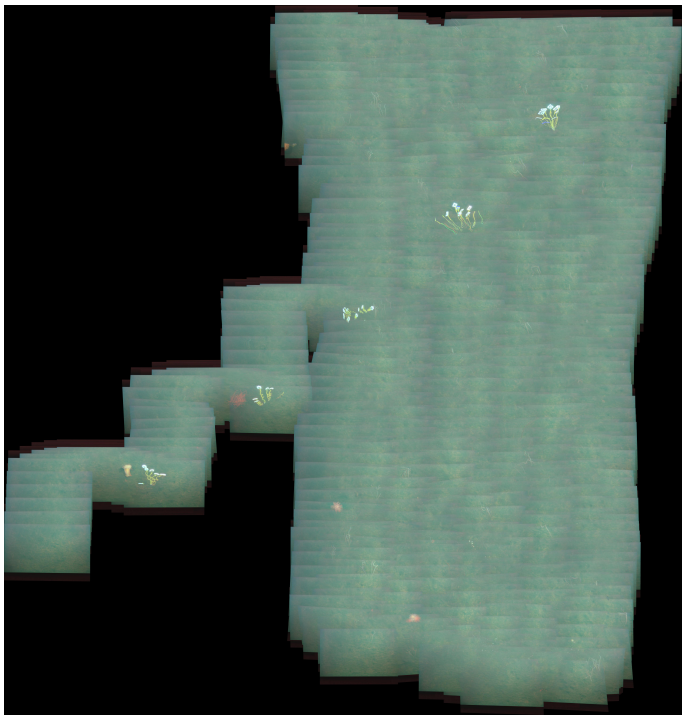

Figure 12. Second half of mapping run

failures and then implement logic to reduce model complexity or omit or omit bad data, the author believes it will be possible to create maps largely without the need for human oversight.

As mentioned earlier, switching the criterion for matching images away from RANSAC in favor of a Hough-like histogram based model for matching, may improve robustness of determining matches between frames. This would carry the potential of being robust to cases where there are many false positive matches, as can often occur when looking at a muddy sea floor.

Another technique commonly in practice that could prove fruitful is to assemble submaps and assemble these into the larger whole map. When flying a "lawnmower" pattern, Each swath lends itself to having its parameters estimated through bundle adjustment, then stitching these swaths together later.

The author would also like to investigate applying structure from motion (SfM) techniques to the problem of non-planar mosaic mapping. As stated before, many SfM techniques rely on assumptions that the mosaicking task clearly violates.

Lastly, the author hopes to collaborate with ROV pilots at MBARI to come up with a user-friendly real-time tool to help them ensure full coverage.

References

[1] M. Brown and D. Lowe. Automatic panoramic image stitching using invariant features. *Int. J. Comput. Vision*, 74(1):59–73, 2007.

[2] S. D. Fleischer. Video mosaicking along arbitrary vehicle paths. In *Proceedings of the 1996 Symposium on Autonomous Underwater Vehicle Technology*, 1996.

[3] J. Koenderink and A. van Doorn. Affine structure from motion. *J. Opt. Soc. Am.*, A(8):377–385, 1991.

[4] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60:91–110, November 2004.

[5] P. H. Milne. *Underwater Acoustic Positioning Systems*. Gulf Publishing Co., Houston, 1983.

[6] O. Pizarro and H. Singh. Toward large-area mosaicing for underwater scientific application. *IEEE Journal of Oceanic Engineering*, 28(4):651–672, 2003.

[7] K. Richmond and S. Rock. An operation real-time large-scale visual mosaicking and navigational system. *OCEANS*, pages 1–6, September 2006.

[8] A. Vedaldi and B. Fulkerson. *VLFeat: An Open and Portable Library of Computer Vision Algorithms*. www.vlfeat.org, 2008.