Comparison of Feature Detection Techniques for AUV Navigation Along a Trained Route

Peter King
Marine Environmental Lab for Intelligent Vehicles (MERLIN)
Memorial University of Newfoundland
St. John's, Canada
peter.king@mun.ca

Benjamin Anstey
Faculty of Engineering & Applied Science
Memorial University of Newfoundland
St. John's, Canada
u96bja@mun.ca

Andrew Vardy
Department of Computer Science / Faculty of Engineering & Applied Science
Memorial University of Newfoundland
St. John's, Canada
av@mun.ca

Abstract

Autonomous underwater vehicles (AUV)s traversing a path will incur positional error drift over time while submerged. We are developing a route following system which is based upon features extracted from the seabed using sidescan sonar collected in a training phase. Through matching of sonar images, this system navigates over a path without the need for a continual global position estimate. At the core of this system is the need to reliably extract features and match images derived from the sonar. At our disposal is an array of algorithms which implement the OpenCV common interface for feature extraction and matching. Using pre-collected sets of data we compare the performance of several of these algorithms in the context of matching sonar image tiles. Our results compare the performance of various feature types over two common sets of data. The feature types tested include SIFT[12], SURF[3], MSER[13], STAR[1], ORB[15], and BRIEF[4].

I. Introduction

For many applications of autonomous underwater vehicles (AUVs) there is a need to traverse a path repeatedly in order to look for changes in the seabed. Changes of interest could include visible changes to the health of benthic habitats, the introduction of security threats, or damage caused to man-made structures such as pipelines. Even AUVs equipped with the most capable inertial navigation systems and Doppler velocity logs will inevitably lose track of their position if they stay submerged for long periods of time [10]. Therefore, without sensor-based feedback, the vehicle will inevitably drift from its nominal trajectory. We are developing an AUV route following system based on features extracted from side-scan sonar images of the seabed collected during training. Many different feature types have been proposed by the computer vision and image processing communities. In this paper we test a variety of different feature types which are extracted from side-scan sonar images and used to determine the AUV's position along a trained route with the help of the discrete Bayes filter.

A. Qualitative Navigation System

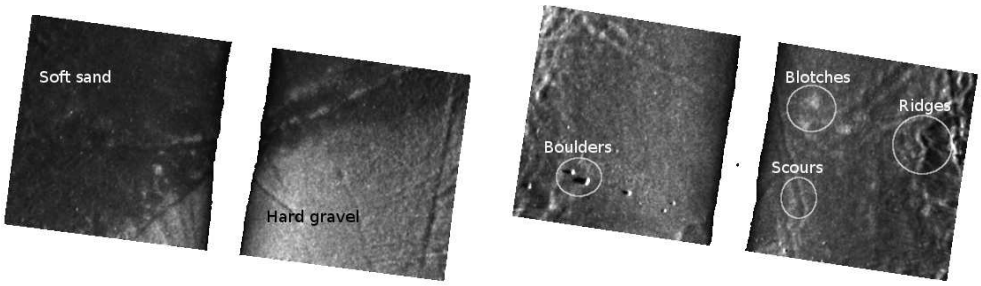
The Qualitative Navigation System (QNS) we are developing is a framework consisting of sonar image generation, feature extraction, path localization and vehicle navigation. This framework supports an algorithm for the registration of sidescan sonar images, which is at the core of our autonomous route following strategy [16]. The approach taken is to represent the route as a sequence of nodes, each with an associated sidescan sonar image tile collected during a training phase. This strategy is based on the notion of topological navigation pioneered in mobile robotics [9] which has also been described as *qualitative navigation* [6], [5]. The approach of qualitative navigation is to represent the environment as a set of connected places with mechanisms for travelling between places. It is quite explicit that these places are *not* represented in the same global coordinate system. These ideas have led to considerable success in recent years in allowing an outdoor mobile robot to autonomously follow a trained route, despite changes in illumination and variable terrain [5], [17]. Our work represents the first attempt to apply a qualitative navigation approach to the underwater domain.

B. Sidescan Image Generation

Sidescan images are built by layering the time series of return intensities from individual sonar pings. This layering utilizes motion data from the vehicle to provide a geometrically correct representation of the sea bottom [8]. Return intensity is affected by sea bottom composition and topography. Figure 1 shows some common intensity variations seen in sidescan data.

C. Image Feature Extraction and Matching

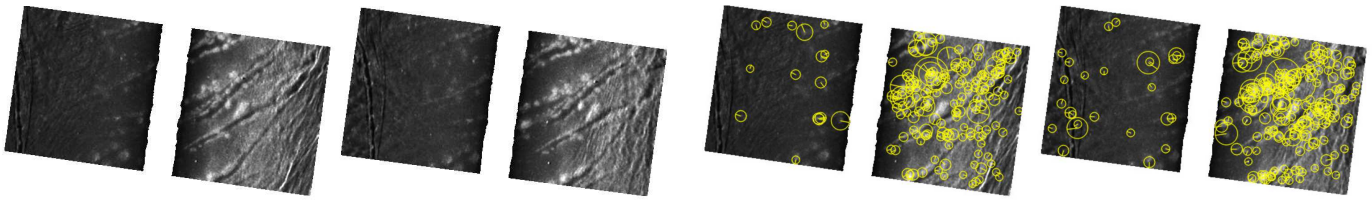
In computer vision and image processing the concept of feature detection refers to methods that aim at computing abstractions of image information and making local decisions at every image point whether there is an image feature of a given type at that point or not. QNS denotes these interesting feature areas as *KeyPoint*s. Once these *KeyPoint*s have been detected, a local image



(a) Hard and soft bottom types

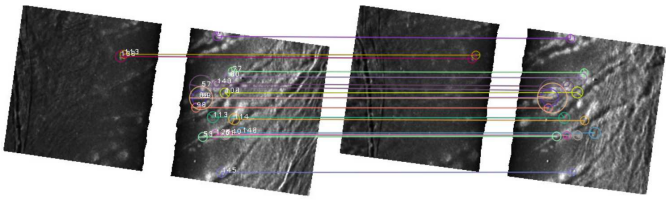
(b) Common characteristic patterns

Fig. 1: Characteristic patterns in side-scan sonar images.



(a) Original sonar tiles

(b) KeyPoints shown by circles: radius is size of neighbourhood; enclosed line indicates orientation



(c) Lines indicate KeyPoint match pairings considered inliers

Fig. 2: Sonar tiles and feature matching

patch around the feature can be extracted. From this patch we extract a description vector denoted as a *descriptor*. A common approach for generating descriptors is based on histograms of the image gradient function. If these histograms are constructed and stored relative to the scale and dominant orientation of the *KeyPoint* then the resulting descriptor can be considered scale and orientation invariant [12].

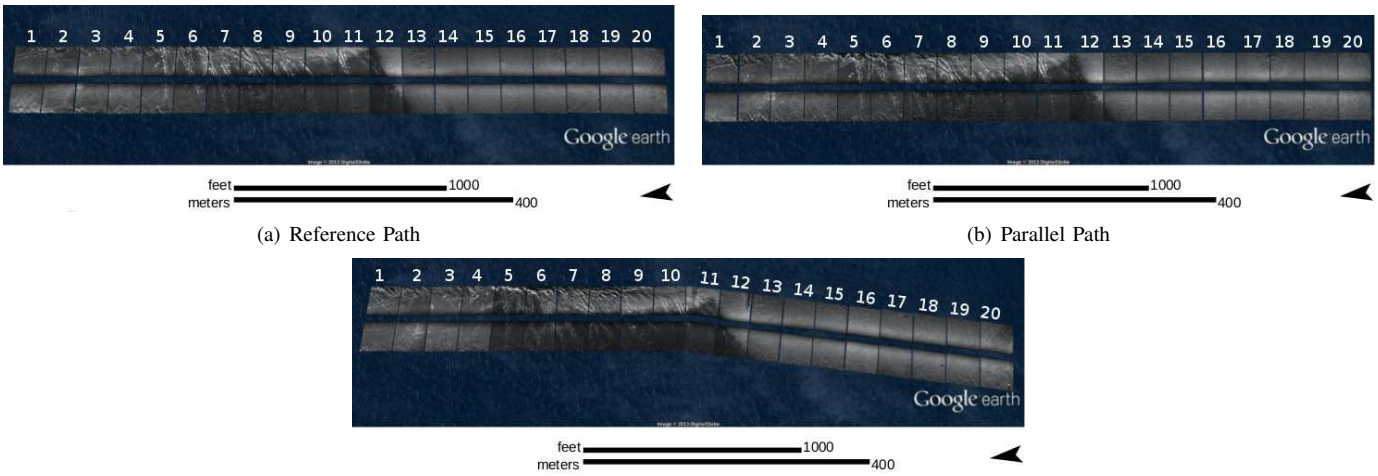
QNS has been designed to support any feature extraction/detection algorithm which implements a common interface as defined by OpenCV [14]. OpenCV has base structures for *KeyPoint* and *descriptor* extraction, as well as matching between image descriptors. This allows the use of stock implementations that were contributed to the OpenCV library or custom implementations. Either a single algorithm or a combination of algorithms can be used simultaneously.

Once *KeyPoint*s and corresponding descriptors have been extracted, they are compared between images to search for matches. Initially there will be many matching descriptors between images. Simple filtering methods can be used to look for agreement in matches and filter out those that are most likely false. The remaining matches are considered inliers and are used for further tests on overall image match.

Based on the number of matches and the consistency of their error vectors a decision can be made as to whether there exists sufficient commonality for the images to be considered a positive match—meaning that they capture the same area of seabed. Figure 2 shows two sonar images, which we refer to as tiles, their extracted *KeyPoint*s and the pairs of matching *descriptors* between them. This process of matching one tile to another is a form of image registration.

II. Test Setup

Using QNS in offline mode, in which previously collected sonar data is utilized instead of real-time data, several feature extraction algorithms were utilized to perform tile matching against a reference set. For each attempted cycle of tile matching and localization, results indicate how strongly each algorithm makes a match, if any, by the resulting count of *KeyPoint* match inliers, as well as the computed average translation to the matched reference path tile. Each algorithm can either make a correct match, no match, or an incorrect match.



(a) Reference Path

(b) Parallel Path

(c) Cross-over Path

Fig. 3: Sonar data used in tests

The test data set was collected in Holyrood, Newfoundland, Canada, in 2013 using Memorial University's Explorer AUV [7]. Sidescan imagery data were collected from an Edgetech 2200M combined sidescan system operating at 400kHz. Presented here are two runs, the first is a track parallel to the reference path, with a slight offset, the second is a track which crosses over the reference track. Figure 3 shows the tracks. The numbers indicate the index of each tile. The paths align such that tile indexes match between tracks, for example tile 4 in the parallel track should match to tile 4 in the reference path.

As described in Section I-B there are common bottom characteristics seen in sidescan sonar data. From the test paths we can abstract the track into three distinct types:

1) Hard sediment with scours (Tiles 1 through 9)
2) Mixed sediment, light and dark regions, with scours (Tiles 10 through 13)
3) Hard sediment with rocks (Tiles 14 through 20)

During image generation, it was assumed that the vehicle's pitch and roll are constant and variations are negligible. Raw sonar data is transformed to a single coordinate system relative to the vehicle's viewpoint. To minimize the effect of scale variations corresponding to variation in vehicle altitude, plotted pings are corrected for slant range. These corrections mean that path and reference images are always oriented North up. Even though there should be minimal variance in scale and orientation, scale and rotation invariant feature detection/extraction algorithms are still employed to increase the system's robustness. *KeyPoint* matches that differ in scale or orientation outside of a certain threshold are rejected.

Measurements are performed via image registration. *KeyPoint*s and *descriptors* are computed for each reference tile and for each *PathTile* as it is collected. The *PathTile descriptors* are matched against each reference tile. To measure the performance of an algorithm several metrics are recorded:

- The total number of features detected
- The number of inlier *KeyPoint* matches obtained after matching and match filtering stages
- The re-projection error associated with re-projecting *PathTile KeyPoint*s onto a ReferenceTile
- The computed navigation translations
- The elapsed time for processing a tile
- The overall performance of localization, represented by a histogram of belief for each location over the reference path, as calculated by a Bayes filter

For this work we focus on the number of inliers as this is the main criterion for tile matching.

A. Algorithms

Using the common interface provided by OpenCV, our system can employ any feature extraction and matching algorithm which utilizes this interface. The algorithm implementations used in this test were all from the OpenCV Feature2d module. A brief description of each is provided here, but the reader is encouraged to refer to the cited papers, as well as the OpenCV documentation, for a more in-depth description.

SIFT [12] (Scale-Invariant feature transform) is a scale invariant feature detection/extraction algorithm. KeyPoint locations are defined as maxima and minima from an applied difference of Gaussians function applied in scale space to a series of smoothed and re-sampled images. Low contrast candidate points and edge response points along an edge are discarded. Dominant orientations are assigned to localized keypoints. SIFT descriptors are obtained by considering pixels around a radius of the key location.

SURF [3] (Speeded Up Robust Features) is a robust local feature detector inspired by SIFT. The standard version of SURF is claimed by its authors to be several times faster than SIFT and more robust against different image transformations. SURF is based on sums of 2D Haar wavelet responses and employs box filters using integral images, as opposed to the more computationally-demanding Gaussian filters employed by SIFT.

MSER [13] (maximally stable external regions) is an interest region detector originally described by Matas et al in 2002. What separates MSER from many other interest region detectors is that it operates on the input image directly without any smoothing, which results in the detection of both fine and coarse structures. MSER utilizes a technique based around merging connected areas based on intensity.

STAR [1] is a modified version of the CenSurE (Center Surrounded Extremas) algorithm. CenSurE features are computed at the extrema of the center-surround filters over multiple scales, using the original image resolution for each scale. They are an approximation to the scale-space Laplacian of Gaussian and can be computed in real time using integral images. CenSurE features are reported to be efficient, distinctive, stable and repeatable in changes of viewpoint.

ORB [15] (Oriented FAST and Rotated BRIEF) is a computationally efficient replacement to SIFT that has similar matching performance, is less affected by image noise, and is capable of being used for real-time performance. ORB builds on the well-known FAST keypoint detector.

BRIEF [4] (Binary Robust Independent Elementary Features) is a binary feature descriptor. Unlike SIFT or SURF, it uses a relatively small number of bits to describe a feature and matches can be evaluated by computing Hamming distance, which is more efficient to compute than the L_2 norm. BREIF allows fast computing and matching of descriptors, but is sensitive to image rotation and scaling.

FREAK [2] (Fast Retina Keypoint) is inspired by the human visual system. Binary strings are computer by comparing image intensity over retinal sampling patterns. FREAK is stated to be efficient in memory and robust and well suited for embedded applications.

III. Test Results

This section presents the numeric results for both the parallel and crossing path tests. The table headings are:

PathTile
: The index of the current path tile

match
: The index of the reference tile(s) which are considered a match

Algorithm
: Name of the algorithm(s) which were able to discern a match

inliers
: Number of keypoint matches considered inliers

avg. reproj. error
: Average error when computed translations are used to reproject matched features together

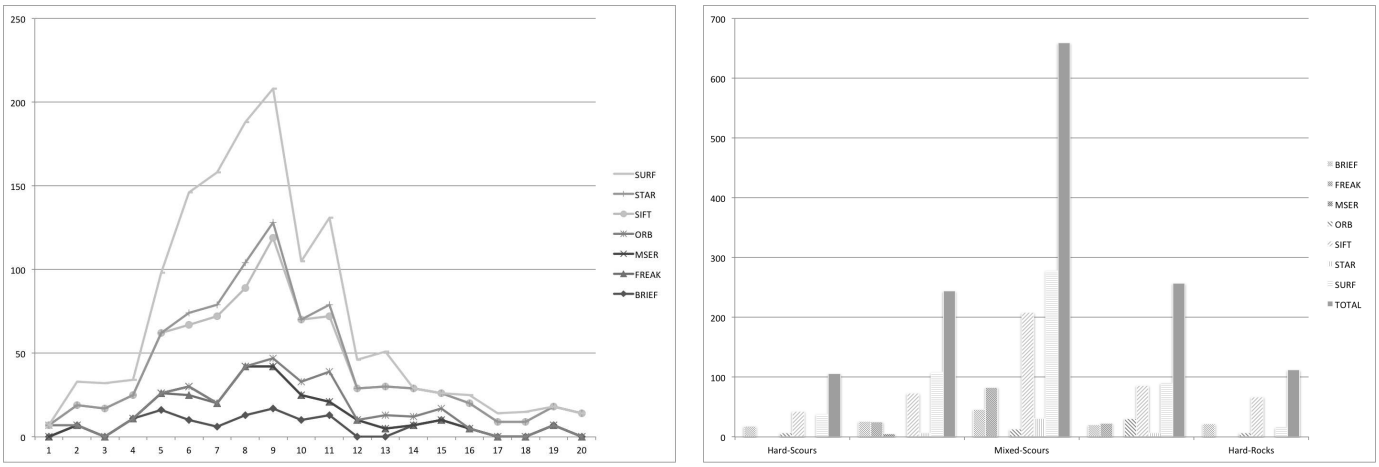
Translation
: The x-y displacement from the path tile image to the reference set image in metres

A. Parallel Test

Table I shows matching results for the paths shown in Figure 3(a) and 3(b).

TABLE I: Matches for parallel test case

PathTile	match	Algorithm	# inliers	avg. reproj. error (m)	Translation (m)
#1	#1	ORB	7	2.58026	(20.736, 22.464)
#2	#1	SIFT	8	3.08309	(5.63293, 2.20557)
	#2	BRIEF	7	3.64914	(21.6252, 24.9398)
		SIFT	12	1.86534	(21.6778, 27.4503)
		SURF	14	1.64289	(21.5534, 27.0679)
#3	#3	SIFT	17	1.73392	(22.0197, 20.3875)
		SURF	15	2.69135	(22.7122, 20.0333)
#4	#4	BRIEF	11	2.29078	(22.7594, 14.1678)
		SIFT	14	1.72482	(22.7886, 14.873)
		SURF	9	3.21908	(23.8942, 17.7874)
#5	#5	BRIEF	16	2.87049	(24.6185, 8.84584)
		FREAK	10	2.50298	(23.4003, 10.2038)
		SIFT	36	1.58227	(23.0788, 9.31071)
		SURF	36	1.90277	(23.6268, 9.08977)
#6	#6	BRIEF	10	2.68559	(25.6846, 8)
		FREAK	15	3.16229	(27.0854, 10.0187)
		MSER	5	1.20102	(25.1292, 9.00558)
		SIFT	37	2.19659	(24.9047, 9.4072)
		STAR	7	1.51093	(26, 10)
		SURF	72	2.25669	(26.5802, 10.5541)
#7	#7	BRIEF	6	2.12379	(25.9694, 3.10136)
		FREAK	14	1.56363	(25.1769, 3.87378)
		SIFT	52	2.18065	(26.2131, 3.47604)
		STAR	7	1.36807	(24, 3)
		SURF	79	2.3006	(26.1499, 2.23769)
#8	#8	BRIEF	13	2.499	(25.3983, 3.00154)
		FREAK	29	1.59642	(25.4033, 2.05421)
		SIFT	47	2.05512	(25.3226, 1.59645)
		STAR	15	1.85625	(25, 1)
		SURF	84	2.06558	(25.318, 1.69534)
#9	#9	BRIEF	17	2.21528	(23.9326, 1.49826)
		FREAK	25	1.95596	(24.786, 2.14274)
		ORB	5	1.96247	(25.92, 3.45599)
		SIFT	72	1.79353	(25.4884, 0.943268)
		STAR	9	2.70111	(24, 1)
		SURF	80	2.34159	(27.1161, 0.75296)
#10	#10	BRIEF	10	2.42383	(26.0903, -4.10565)
		FREAK	15	2.54538	(26.4529, -3.30032)
		ORB	8	3.32167	(25.92, -2.88)
		SIFT	37	1.92997	(26.5547, -5.10419)
		SURF	35	2.38672	(26.3319, -5.36983)
#11	#11	BRIEF	13	2.48528	(27.3695, -3.25287)
		FREAK	8	2.13787	(27.7129, -4.0283)
		ORB	18	1.78359	(27, -3)
		SIFT	33	1.64278	(27.0669, -3.05798)
		STAR	7	1.7655	(28, -4)
		SURF	52	1.52498	(27.802, -3.46471)
#12	#12	FREAK	10	2.32828	(28.5747, -5.00645)
		SIFT	19	2.01879	(28.2564, -2.82724)
		SURF	17	2.00337	(28.4929, -5.19972)
#13	#13	FREAK	5	2.48475	(30.233, -13.793)
		ORB	8	1.7734	(30, -13)
		SIFT	17	1.61094	(30.2646, -13.337)
		SURF	21	2.68189	(28.6983, -12.1878)
#14	#14	BRIEF	7	1.30569	(33, -8.03935)
		ORB	5	0.778263	(31.6, -7.8)
		SIFT	17	1.50406	(32.2049, -8.29949)
#15	#15	BRIEF	10	1.81741	(33.6329, -14.4245)
		ORB	7	1.64752	(33, -12)
		SIFT	9	1.9683	(33.908, -14.1532)
#16	#16	BRIEF	5	2.02516	(32.9323, -18.3128)
		SIFT	15	1.73278	(33.0257, -17.2754)
		SURF	5	2.26771	(33.947, -17.2622)
#17	#17	SIFT	9	2.1641	(33.8209, -22.1529)
		SURF	5	1.94851	(32.7986, -20.1072)
#18	#18	SIFT	9	1.04182	(34.1425, -24.998)



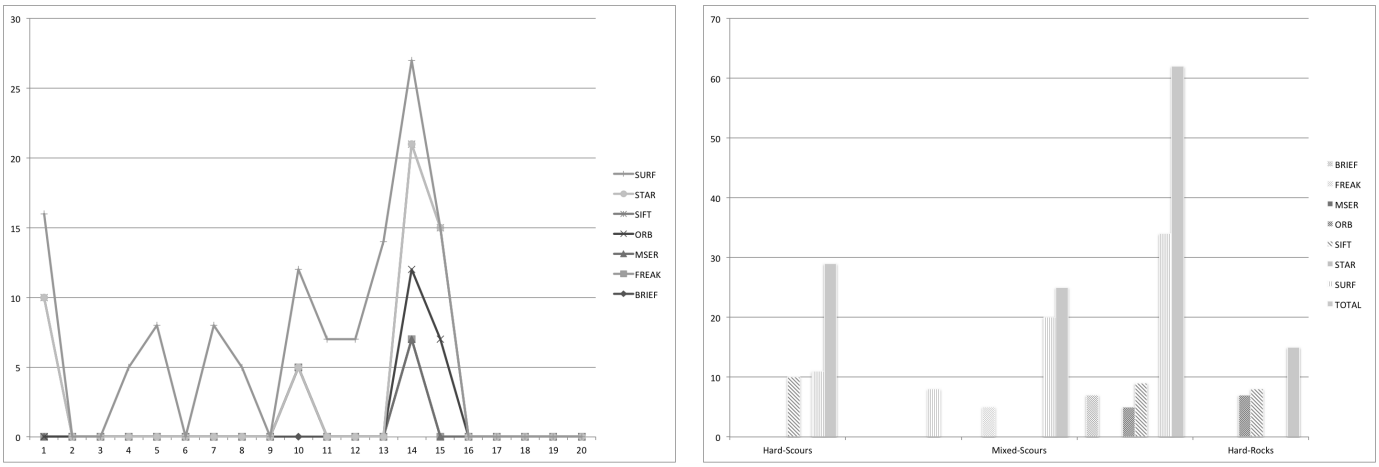
(a) Inliers for each reference tile

(b) Inliers for each bottom type

Fig. 4: Inliers from parallel test

PathTile	match	Algorithm	# inliers	avg. reproj. error (m)	Translation (m)
		SURF	6	2.13888	(35.6597, -26.3824)
#19	#19	BRIEF	7	2.1856	(34.2087, -28.6082)
		SIFT	11	1.51405	(35.4281, -29.1443)
#20	#20	SIFT	14	1.41055	(37.0938, -30.246)

Overall we can see that SIFT is the dominant algorithm, providing successful matches for 19 of the 20 tiles, with a single false positive. If we consider the contribution of all algorithms there are successful matches in all 20 match cases. Figure 4(a) shows the distribution of inliers over all tiles in the reference set. SIFT and SURF emerge as overall dominant schemes in this context. As previously stated, the data has distinct regions. Figure 4(b) shows inlier counts for each of the three described regions and for areas between distinct regions which can be considered transition zones.



(a) Inliers for each reference tile

(b) Inliers for each bottom type

Fig. 5: Inliers from crossing test

B. *Crossing Test*

TABLE III: Matches for crossing test case

PathTile	match	Algorithm	# inliers	avg. reproj. error (pixels)	Translation (pixels)
#1	#1	SIFT	10	2.39784	(79.1746, 80.4562)
		SURF	6	1.91712	(79.1113, 82.3723)
#4	#4	SURF	5	2.69148	(84.8333, 77.5135)
#5	#5	SURF	8	1.81729	(83.4887, 71.6707)
#7	#7	SURF	8	1.65467	(85.3678, 69.8228)
#8	#8	SURF	5	3.83184	(85.4777, 62.1295)
#10	#10	FREAK	5	2.19048	(83.6851, 59.3588)
		SURF	7	2.74446	(83.5969, 61.135)
#11	#11	SURF	7	2.44742	(69.5806, 63.258)
#12	#12	SURF	7	2.62761	(41.0684, 28.4642)
#13	#13	SURF	14	2.52666	(13.9366, 13.5482)
#14	#14	BRIEF	7	2.57627	(-14.0078, 15.5366)
		ORB	5	1.88666	(-13, 15)
		SIFT	9	1.92271	(-13.4786, 14.2395)
		SURF	6	2.3531	(-13.5469, 15.1193)
#15	#15	ORB	7	2.44619	(-43.2, 2.4)
		SIFT	8	1.4561	(-42.2753, 2.93164)

The crossing test begins with less overlap with the reference set than does the parallel test and we see the effect of this on the reduced inlier count in the first half of the path. In the second half the track begins to converge with the reference path. Overall we see 11 out of a possible 12 tiles matching, with SURF being the dominant contributor. We can also note the complete lack of matches in the final 5 tiles. Figure 5(a) shows the distribution of inliers in this case.

We again plot inliers over the three bottom types and transition zones, shown in Figure 5(b). The figure can be misleading as the area approaching the third bottom type appears to be the strongest, though the majority of it produced no matches. As the track converges to the reference path we see an increase in inliers, but as we travel further into the third bottom type, the number of inliers begin to reduce, as in the parallel case.

IV. Conclusions

The results presented in this paper have shown how readily available feature extraction and matching algorithms can be used to perform matching on image tiles generated from sidescan sonar. It is clearly shown that the various techniques perform quite differently in the two cases presented here. In the first case SIFT and SURF perform quite well and provide more than sufficient matching for navigation. FREAK, BRIEF, and ORB performed reasonably in the first two-thirds of the track, with MSER only providing a single match. In the second case SURF performed reasonably in the first two-thirds of the track with little contribution from the remaining algorithms.

If we look at instances where the top performing algorithms failed we find that other algorithms obtained correct matches. For tile 2 of the parallel path SIFT shows a false positive, but SURF and BRIEF are correct. In tile 15 of the cross track, SURF fails to make a match, but ORB and SIFT do. As a means to improve robustness a multi-detector approach would be recommended. In both cases a simple majority vote would improve matches from 18 to 20 in the parallel case, and from 10 to 11 in the cross case.

For both test cases the performance deteriorates as we enter the region of hard sediment and rocks. This region has very little texture and only isolated point-like variations. None of the selected algorithms worked well in this region. In the parallel case SIFT and SURF functioned, but with lower inlier counts. In the crossing case no algorithm provided matches in the last 5 tiles. A new detector for this region type could be developed which focuses on these intermittent point like features. If targeted towards this type of terrain, such a detector could operate in parallel with other detectors suited for mixed sediment and scours, yielding a more robust system. Given the open framework of OpenCV employed by the QNS system, this will be an area of focus in the near term.

Acknowledgment

The work presented in this paper was funded by the Atlantic Canada Opportunities Agency, through an Atlantic Innovation Fund grant, under the Responsive AUV Localization and Mapping project (REALM), and Research and Development Corporation of Newfoundland and Labrador. The authors would also like to acknowledge the work of Jason Gedge in the development of the QNS software.

References

[1] M. Agrawal et al., "CenSurE: Center Surround Extremas for Realtime Feature Detection and Matching," Computer Vision (ECCV), 2008, pp.102–115

[2] A. Alahi et al., "FREAK: Fast Retina Keypoint," Computer Vision and Pattern Recognition (CVPR), pp.510–517, 2012.

[3] H. Bay et al., "Surf: Speeded Up Robust Features," Computer Vision and Image Understanding, 2008, pp.346–359

[4] M. Calonder et al., "BRIEF: Binary Robust Independent Elementary Features," Computer Vision (ECCV), 2010, pp.778–792

[5] Z. Chen and S. Birchfield, Qualitative vision-based path following, IEEE Transactions on Robotics, vol. 25, no. 3, pp. 749754, 2009.

[6] D. Dai and D. Lawton, Range-free qualitative navigation, in IEEE ICRA, 1993.

[7] International Submarine Engineering. (July, 2013) AUVs. [Online]. http://www.ise.bc.ca/auv.html

[8] P. King, A. Vardy, P. Vandrish, and B. Anstey, "Real-time side scan image generation and registration framework for AUV route flollowing," Proc. Autonomous Underwater Vehicles, 2012.

[9] B. Kuipers and Y.-T. Byun, A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations, Journal of Robotics and Autonomous Systems, vol. 8, pp. 4763, 1991.

[10] J. J. Leonard, A. A. Bennett, C. M. Smith, and H. J. S. Feder, Autonomous underwater vehicle navigation, Dept. Ocean Eng., MIT Marine Robot. Lab., Cambridge, MA, Tech. Memorandum, Jan. 1998.

[11] S. Leutenegger, S. et al., "BRISK: Binary robust invariant scalable keypoints," Computer Vision (ICCV), 2011, pp.2548–2555

[12] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," Computer Vision and Image Understanding, 2004, pp.91–110

[13] J. Matas et al., "Robust wide baseline stereo from maximally stable extremal regions," Proc. British Machine Vision Conference, 2002, pp.384–396

[14] G. Bradski, "The OpenCV Library", Dr. Dobb's Journal of Software Tools", 2000.

[15] E. Rublee et al., "ORB: an efcient alternative to SIFT or SURF," Proc. IEEE Intl. Conf. on Computer Vision (ICCV), 2011

[16] P. Vandrish, A. Vardy, and P. King, "Towards AUV Route Following Using Qualitative Navigation," Proc. Computer and Robot Vision, 2012.

[17] A. Zhang and L. Kleeman, Robust appearance based visual route following for navigation in large-scale outdoor environments, The International Journal of Robotics Research, vol. 28, no. 3, pp. 331356, 2009.