Feature Tracking Evaluation for Pose Estimation in Underwater Environments

Florian Shkurti, Ioannis Rekleitis, and Gregory Dudek
School of Computer Science
McGill University, Montreal, QC, Canada
{*florian, yiannis, dudek*}*@cim.mcgill.ca*

***Abstract*—In this paper we present the computer vision component of a 6DOF pose estimation algorithm to be used by an underwater robot. Our goal is to evaluate which feature trackers enable us to accurately estimate the 3D positions of features, as quickly as possible. To this end, we perform an evaluation of available detectors, descriptors, and matching schemes, over different underwater datasets. We are interested in identifying combinations in this search space that are suitable for use in structure from motion algorithms, and more generally, vision-aided localization algorithms that use a monocular camera. Our evaluation includes frame-by-frame statistics of desired attributes, as well as measures of robustness expressed as the length of tracked features. We compare the fit of each combination based on the following attributes: number of extracted keypoints per frame, length of feature tracks, average tracking time per frame, number of false positive matches between frames. Several datasets were used, collected in different underwater locations and under different lighting and visibility conditions.**

***Keywords*-Underwater feature tracking; Sensor fusion; Mapping; State Estimation; Performance evaluation techniques;**

I. Introduction

The task of localization is one of the fundamental problems in mobile robotics. It is particularly important in any scenario where robots are required to exhibit autonomous behavior, since many navigation and mapping algorithms require, or benefit from, the existence of some prior belief about the robot's pose. In the underwater domain the problem of localization is particularly challenging as it is a GPS denied, highly unstructured natural environment, without man-made landmarks. This paper presents an overview of a six degree of freedom (6DOF) state estimation algorithm for an underwater vehicle, and in particular, it presents an analysis of the computer vision components that it relies on. The described algorithm combines measurements from an Inertial Measurement Unit (IMU) sensor and a camera in order to accurately track the pose of the vehicle. An essential part of this algorithm is feature tracking using a monocular camera, and estimation of 3D feature positions with respect to the camera frame. Our goal in this paper is to evaluate which feature detectors, descriptors, and matching strategies are most suitable for integration with the rest of the state estimation algorithm. While our focus is directed towards this particular algorithm, our evaluation is sufficiently informative for other closely related problems, such as visual odometry and structure from motion.

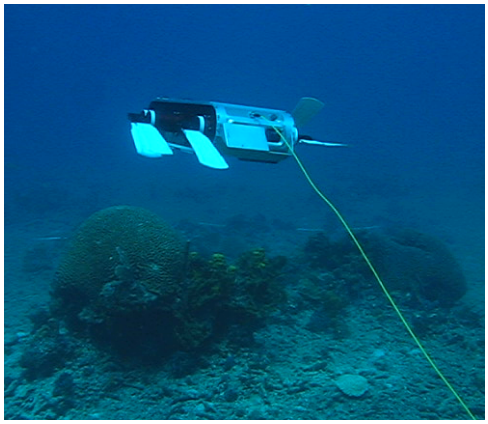


Figure 1. The Aqua vehicle starting a dataset collection run

The target vehicle is a robot belonging to the Aqua family [1] of amphibious robots; see Fig. 1. It is a six-legged vehicle capable of reaching depths of 35 m in remotely-controlled or autonomous operation. It is equipped with three IEEE-1394 IIDC cameras and an IMU. One of the primary uses of this vehicle is to conduct visual mapping over coral reefs. The state estimation algorithm, first proposed by Mourikis and Roumeliotis [2] uses visual input from a downward-facing camera to correct the errors resulting from the double integration of the IMU signal.

II. Related Work

Corke et. al [3] presented experimental results of an underwater robot performing stereo visual odometry using the Harris feature detector and the normalized cross-correlation similarity measure (ZNCC). They reported that, after outlier rejection, 10 to 50 features were tracked from frame to frame, and those were sufficient for the stereo reconstruction of the robot's 3D trajectory, which was a square of area 30m by 30m.

Barfoot [4] described a 6DOF pose estimation technique, adapted from FastSLAM 2.0 [5], whose input was the odometry and stereo images of a terrestrial mobile robot. SIFT features were used as observations in the filter and were matched via best-bin-first search in a kd-tree. The author demonstrated through field experiments that the proposed approach led to 0.5% to 4% localization errors over distances of 40-120 meters.

Newcombe and Davison [6], as well as Klein and Murray [7], have presented online, accurate 3D reconstruction of small environments using a monocular camera. Their tracking system uses the FAST detector and performs search along epipolar lines in order to find matching pixel neighborhoods.

The landscape of performance evaluations of feature detectors and descriptors is certainly rich, as there has been extensive related work in trying to experimentally assess the degree of invariance of said detectors to affine transformations. For example, Mikolajczyk and Schmid [8] compared the following detectors: Harris corners, Harris-Laplace, Hessian-Laplace, Harris-Affine and Hessian-Affine, based on their repeatability, in other words, their ability to consistently detect the same keypoints in two different images of the same scene. As for feature descriptors, their evaluation included SIFT [9], Gradient Location and Orientation Histogram (GLOH), cross-correlation of neighborhood pixel values, PCA-SIFT, moment invariants, shape context, steerable filters, spin images, and differential invariants. Some of these descriptors are low dimensional while others are high. The evaluation criterion they used in their experimental setup was precision and recall of feature matches, where a 'true' match was defined as having small vector distance between the corresponding descriptors. Their dataset included scenes that had different lighting and geometric transformations, however, the images were not successive video frames of the same scene, and they did not capture underwater environments. The main conclusion of the above work was that the high dimensional detectors, like GLOH and SIFT were robust and most distinctive, regardless of the detector used.

Closely related to our work is the quantitative comparison of feature extractors for visual SLAM done by Klippenstein and Zhang [10]. Their experimental methodology included comparison of precision and recall curves of the Harris corner detector, the Kanade-Lucas-Tomasi (KLT) tracker [11], and the SIFT detector. Their conclusion was that all detectors could be tuned to perform well for visual SLAM, but in particular, the KLT tracker was better suited for image sequences with small baseline, compared to SIFT.

III. 6DOF State Estimation of an Autonomous Underwater Vehicle

The localization algorithm [2], [12] used is based on the extended Kalman filter (EKF) formulation. It integrates IMU measurements [13], thus providing direct (but noisy) estimates of vehicle dynamics. Furthermore, by performing feature correspondence it estimates motion parameters, and obtains constraints, from the camera data, thereby correcting the drifting IMU integration estimates; see Fig. 2 for a schematic of the algorithm. The innovation of this algorithm is that the state contains the latest pose of the IMU and a list of m camera poses, instead of the vehicle pose and the detected feature positions. As such, this approach does not suffer from the dimensionality explosion common to traditional SLAM algorithms. More formally, the state vector that we want to estimate is the following:

$$x = \begin{bmatrix} q_G^I & b_g & v_I^G & b_a & p_I^G & q_G^{C_1} & p_{C_1}^G & \dots & q_G^{C_m} & p_{C_m}^G \end{bmatrix} \quad (1)$$

where q_G^I is the quaternion that expresses the rotation from a global frame of reference to the IMU frame, which is attached on the robot [1], v_I^G is the velocity, and p_I^G is the position of the IMU frame in coordinates of the global frame. b_g and b_a are the bias vectors of the gyroscope and the accelerometer, respectively. The pairs $\{q_G^{C_i} \; p_{C_i}^G\}$ represent the transformations between the global frame and the i-th camera frame. The number m of such transformations embedded in the state vector is bounded above for practical purposes, and because features are expected to be tracked in a relatively small number of images.

The main idea behind fusing information from these two sources is that, provided we are able to track features in the world and correctly estimate their 3D position with respect to the camera, we can sufficiently correct the drifting IMU pose estimates, which are obtained by integration; More specifically, the above vector will be modified in three phases: (i) The propagation phase, where IMU measurements are integrated so as to predict the IMU pose. (ii) The augmentation phase, which occurs when an image is recorded. The transformation from the global frame to the camera frame is appended to the right of the state vector. (iii) The update phase, which occurs whenever a feature track terminates. Then we can estimate its 3D position and correct the state vector, specifically the IMU integration drift, based on the constraints set by the motion of the feature.

In this paper we focus our attention on (iii), and in particular, on the vision component of the system. Our goal is to evaluate which detectors, descriptors, and matching strategies enable us to estimate the 3D positions of features, as quickly and accurately as possible, using a monocular camera. We aim to use this estimator in underwater environments, which are generally more challenging than most of their indoor counterparts for the following reasons:

(a) *They are prone to rapid changes in lighting conditions*. The canonical example that illustrates this is the presence of caustic patterns, which are due to refraction, and the non-uniform reflection, and penetration of light when it transitions from air to the rippling surface of the water.

(b) *They often have limited visibility*. This is due to many factors, some of which include light scattering from suspended plankton and other matter, which cause blurring and "snow effects". Another reason involves the incident

[1]Recall that quaternions provide a well-behaved parameterization of rotation. We are following the JPL formulation of quaternions; see [13], [14]

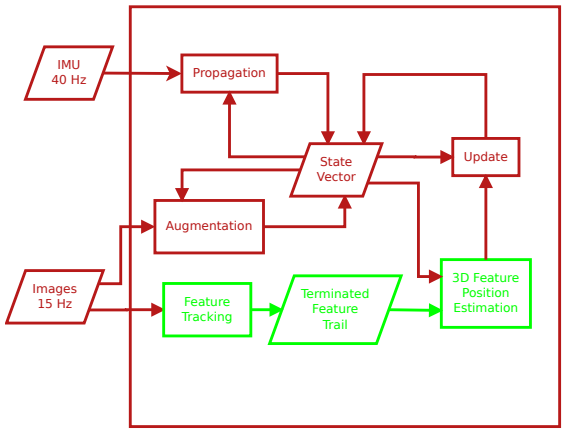


Figure 2. An overview of the 6DOF state estimation algorithm. The state vector appears in Eq. (1). The green-colored components are examined in this paper.

angle at which light rays hit the surface of the water. Smaller angles lead to less visibility.

(c) *They impose loss of contrast and colour information with depth*. As light travels at increasing depths, different parts of its spectrum are absorbed. Red is the first color that is seen as black, and eventually orange, yellow, green and blue follow. This absorption sequence refers to clear water, and is not necessarily true for other types.

A. Feature Detectors And Descriptors

Our evaluation compares the following feature detectors: SURF, Shi-Tomasi, FAST, CenSurE, and the SURF descriptor.

The SURF [15] detector is designed to be scale and (partially) rotation invariant. It detects keypoints at local extrema of the determinant of the image's approximate Hessian. This filter response is evaluated at different scales so as to achieve scale invariance. The scale space is divided into overlapping *octaves*, each of which consists of filter responses at increasing scales. Rotation invariance is due to the assignment of a dominant orientation of Haar wavelet responses around the detected keypoint. Each keypoint is characterized by a SURF descriptor, which is a vector that consists of sums of Haar wavelet responses around its oriented neighborhood. Neither the detector nor the descriptor use any color information.

The Shi-Tomasi detector [16] is invariant under affine transformations, but not under scaling. Its keypoints are image locations where the 2nd moment matrix has two large eigenvalues, indicating image intensity change in two directions, i.e. a corner.

The FAST (Features from Accelerated Segment Test) [17] detector, on the other hand, focuses on speed of keypoint detection rather than robustness to noise and invariance properties. It uses a decision tree to classify a pixel as a keypoint if a circle of pixels around it has arcs that are much brighter or darker than the center. The authors mention that while this classifier is not very robust to noise, scaling and illumination variations, it has high repeatability and rotation invariance.

Finally, the CenSurE (Center Surround Extrema) [18] detector aims to identify keypoints at any scale, without resorting to image subsampling like SIFT, or filter upsampling like SURF. It does this by searching for extrema of the image's approximate Laplacian, using polygonal (e.g. octagonal) center-surround Haar wavelet filters. By comparison, SURF uses box-shaped filters for its descriptor, resulting in less symmetry and partial rotation invariance. Finally, among the detected extrema, only the ones with high Harris response are kept, so as to eliminate keypoints detected along edges, which might be unstable for tracking.

B. Feature Matching

The feature matching component of our evaluation is based on Muja and Lowe's work [19]. More specifically, we match feature pairs based on Approximate Nearest Neighbor search among the respective SURF descriptors. We accept a match if the distance ratio of the nearest neighbor over the second nearest neighbor is below a certain threshold, and if the feature correspondence is bi-directional. For the Approximate Nearest Neighbor Search we experimented with both hierarchical K-means and randomized kd-trees. For combinations that use FAST, CenSurE, or Shi-Tomasi keypoints, we use normalized cross-correlation as a measure of similarity between image patches around said keypoints.

C. Feature Position In Camera Coordinates

Consider a single feature, f, that has been tracked in n consecutive camera frames, $C_1, C_2, ..., C_n$. In the state vector mentioned previously, each camera frame C_i is represented by $\{q_G^{C_i}, p_{C_i}^G\}$ where G is a global reference frame, $q_G^{C_i}$ is the unit quaternion that expresses the rotation from the global frame G to the camera frame C_i, and $p_{C_i}^G$ expresses the position of the origin of camera frame C_i in global frame coordinates.

Now, let us denote $p_f^{C_i} = \begin{bmatrix} X_f^{C_i} & Y_f^{C_i} & Z_f^{C_i} \end{bmatrix}$ to be the 3D position of feature f expressed in camera frame C_i coordinates. This is what we are interested in estimating as accurately as possible. Also, let $R(q_G^{C_i})$ denote the rotation matrix that represents the same rotation expressed by the quaternion $q_G^{C_i}$. Then, we can write the following:

$$R(q_{C_1}^{C_i}) = R(q_G^{C_i})R(q_{C_1}^G) \quad (2)$$

$$p_{C_1}^{C_i} = R(q_G^{C_i})(p_{C_1}^G - p_{C_i}^G) \quad (3)$$

about the transformation between the first frame in which the feature was seen and the rest. So, we have:

$$
\begin{aligned}
p_f^{C_i} &= R(q_{C_1}^{C_i}) p_f^{C_1} + p_{C_1}^{C_i} \qquad (4)\\
&= Z_f^{C_1} \left(R(q_{C_1}^{C_i}) \begin{bmatrix} \frac{X_f^{C_1}}{Z_f^{C_1}} & \frac{Y_f^{C_1}}{Z_f^{C_1}} & 1 \end{bmatrix}^t + \frac{1}{Z_f^{C_1}} p_{C_1}^{C_i} \right)
\end{aligned}
$$

If we let $\alpha_f = \frac{X_f^{C_1}}{Z_f^{C_1}}$, $\beta_f = \frac{Y_f^{C_1}}{Z_f^{C_1}}$, and $\gamma_f = \frac{1}{Z_f^{C_1}}$ then

$$
\begin{aligned}
p_f^{C_i} &= Z_f^{C_1} \left(R(q_{C_1}^{C_i}) \begin{bmatrix} \alpha_f & \beta_f & 1 \end{bmatrix}^t + \gamma_f p_{C_1}^{C_i} \right) \qquad (5)\\
&= Z_f^{C_1} \begin{bmatrix} h_{i1}(\alpha_f, \beta_f, \gamma_f) \\ h_{i2}(\alpha_f, \beta_f, \gamma_f) \\ h_{i3}(\alpha_f, \beta_f, \gamma_f) \end{bmatrix} \qquad (6)
\end{aligned}
$$

Now, assuming a simple pinhole camera model, and disregarding the effects of the camera's calibration matrix, we can model the projection of feature f on the projection plane $Z^{C_i} = 1$ as follows:

$$
z_f^{C_i} = \frac{1}{h_{i3}(\alpha_f, \beta_f, \gamma_f)} \begin{bmatrix} h_{i1}(\alpha_f, \beta_f, \gamma_f) \\ h_{i2}(\alpha_f, \beta_f, \gamma_f) \end{bmatrix} + n_f^{C_i} \qquad (7)
$$

where $z_f^{C_i}$ is the 2×1 measurement, and $n_f^{C_i}$ is noise associated with the process, due to miscalibration of the camera, motion blur, and other factors. If we stack the measurements from all cameras into a single $2n \times 1$ vector z_f and similarly for the projection functions h_i into a $2n \times 1$ vector h_f, then we will have expressed the problem of estimating the 3D position of a feature as a nonlinear least-squares problem with 3 unknowns:

$$
\underset{\alpha_f, \beta_f, \gamma_f}{\operatorname{argmin}} \| z_f - h_f(\alpha_f, \beta_f, \gamma_f) \| \qquad (8)
$$

Provided we have at least 3 measurements of feature f, i.e. provided we track it in at least 3 frames, we use the Levenberg-Marquardt nonlinear optimization algorithm in order to get an estimate $\hat{p}_f^{C_1}$ of the true solution. One problem with this approach is that nonlinear optimization algorithms do not guarantee a global minimum, only a local one, provided they converge. Another problem is that if feature f is recorded around the same pixel location in all the frames in which it appears, then the measurements we will get are going to be linearly dependent, thus providing no information about the depth of f. In other words, feature tracks that have small baseline have to be considered outliers, unless we exploit other local information around the feature to infer its depth. It is worth considering whether we would get better 3D feature position estimates if we treated γ_f as the only unknown in the system, and we fixed $[\alpha_f \quad \beta_f] = z_f^{C_1}$. Our analysis of empirical data suggests that this did not contribute to a significant improvement, and the reason is that Eq. (8) is the optimal solution given the noise characteristics in Eq. (7). Finally, another potential pitfall is that the transformations in Eq. (3) might be themselves noisy, which will also affect the solution of the least squares problem.

D. Outlier Detection

In order to identify false positive matches between two consecutive frames we use epipolar constraints imposed by the fundamental matrix between said frames. We rely on two different methods of estimating the fundamental matrix: one is the RANSAC method [20], which works well as long as there is a significant number of inliers among the matches. The other one is via direct computation [21], which is made possible due to the presence of the IMU:

$$
\begin{aligned}
F &= K^{-t} \left[p_{C_{i+1}}^{C_i} \right]_\times R(q_{C_{i+1}}^{C_i}) K^{-1} \qquad (9)\\
R(q_{C_i}^{C_{i+1}}) &= R(q_G^{C_{i+1}}) R(q_{C_i}^G)\\
p_{C_{i+1}}^{C_i} &= R(q_G^{C_i})(p_{C_{i+1}}^G - p_{C_i}^G)
\end{aligned}
$$

where $[\quad]_\times$ is the cross-product matrix, and K is the calibration matrix of the camera. The latter estimation method is useful as a complement to the former, when the presence of outlier matches is significant. We classify a pair of keypoints, x_1 and x_2 as an outlier match if for either of the two estimates of the fundamental matrix:

$$
|x_2^t F x_1| > \tau \qquad (10)
$$

where τ is a threshold (in our experiments $\tau = 1.0$ pixels). We classify a feature track as an outlier if any of its constituent matching pairs are outliers.

IV. Experimental Results

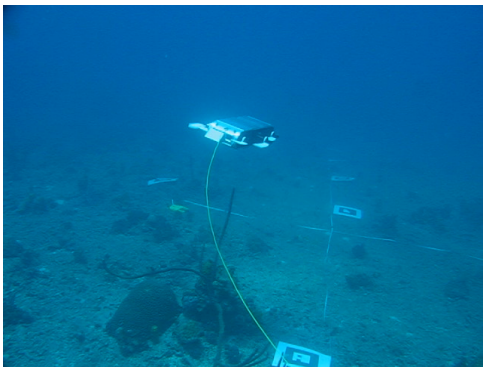


Figure 3. The Aqua vehicle over the marked area.

Our experimental comparison included the following combinations of detectors, descriptors, and matching strategies: SURF keypoints annotated with the SURF descriptor and matched using Approximate Nearest Neighbours, SURF keypoints matched using ZNCC, FAST keypoints matched using ZNCC, CenSurE keypoints matched using ZNCC,

and Shi-Tomasi keypoints matched using ZNCC. For each of the experiments conducted we attempted to tune the parameters of individual combinations in such a way that each of them performs as well as possible. The outlier detection parameters were fixed for all experiments, so that all combinations are treated in the same way.

A. Underwater Camera And IMU Datasets

We rely on 3 different datasets of synchronized camera and IMU input in order to conduct our evaluation. The datasets represent distinct underwater environments that are relatively feature rich, and have been recorded under varying illumination conditions. All datasets were recorded at approximately 30 feet depth. The frame rate of the downward-looking camera was set at 15Hz and each image is (after undistortion and cropping) 870×520 pixels. The camera is a PointGrey Dragonfly Hi-Color 1024×768 pixels. The IMU sensor we used was a MicroStrain 3DM-GX1, and its sampling rate was set at 50Hz. Both sensors are on board of one of the Aqua family of amphibious robots [1]. More specifically:

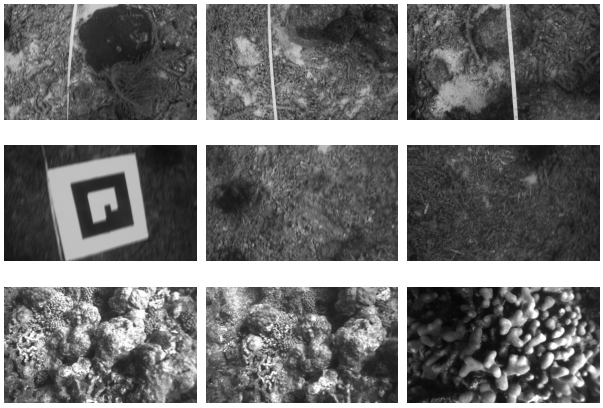


Figure 4. Sample images from each dataset

Dataset1 features a straight 30 meter-long trajectory, where the robot moves at approximately 0.2 meters/sec forward, while preserving its depth. The sea bottom is mostly flat, and the robot moves about 2 meters over it. A white 30 meter-long tape has been placed on the bottom, both to provide ground truth for distance travelled, and to facilitate the straight line trajectory. Its 2600 greyscale camera frames were collected in approximately 3 minutes.

Dataset2 is an L-shaped trajectory, where the robot goes straight for about 7 meters, turns left, and then continues straight for about 3 meters, while preserving depth. Again, the sea bottom is mostly flat, and the robot moves about 2 meters over it, at speed 0.3 meters/sec. The difference compared to *Dataset1* is that in this case we have placed ARToolKit [22] tags on the sea bottom, in order to be able to get estimates of relative rotations and translations between the robot and the tags, which will be useful as ground truth when we try to estimate the 3D positions of the features tracked. Its 600 greyscale images were collected in approximately 45 seconds.

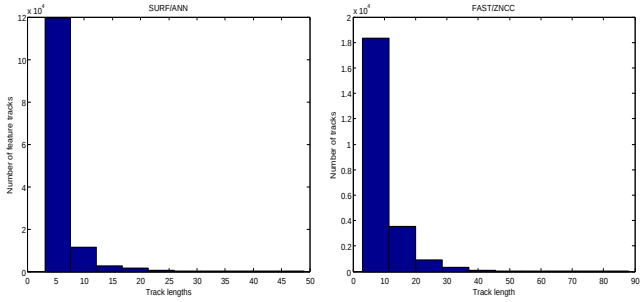


Figure 5. Distribution of feature track lengths for two representative combinations. The figure on the left shows the length distribution for the SURF detector, and Approximate Nearest Neighbor matching. The one on the right shows it for the FAST detector, matched by the normalized cross-correlation score.

Dataset3 features a remotely controlled trajectory over a coral reef, where the robot moves for 20 seconds at speed 0.2m/s. It is the most feature-rich and blur-free among the datasets we have used, it consists of 300 images and it differs from the previous two datasets in that the robot changes depth.

B. Feature Track Lengths

The distribution of feature track lengths is shown in Fig. 5, and it is a decaying curve for all combinations. The SURF detector seems to track most features for an average of 5 camera frames, and it is virtually unable do it for more than 25 frames, using Approximate Nearest Neighbors as the matching method. From the datasets we observed that, given the almost constant forward velocity of the robot, each feature is visible in the camera's field of view for about 25 frames. This means that the combination mentioned above can normally track approximately $1/5^{th}$ of the real trajectory of a feature. Fig. 6 shows that this tracking is very accurate.

C. False Positives

Figure 6 demonstrates the difference in accuracy between the combinations involving SURF, and all the rest. Matching FAST keypoints with the normalized cross-correlation similarity score results in as many as 9% false matches, while the same quantity appears to be 5% for SURF. This difference is pronounced when the image has motion blur, since the normalized cross-correlation similarity measure does not take into account image intensity gradients of any sort, so one should apply very aggressive outlier detection schemes when using it.

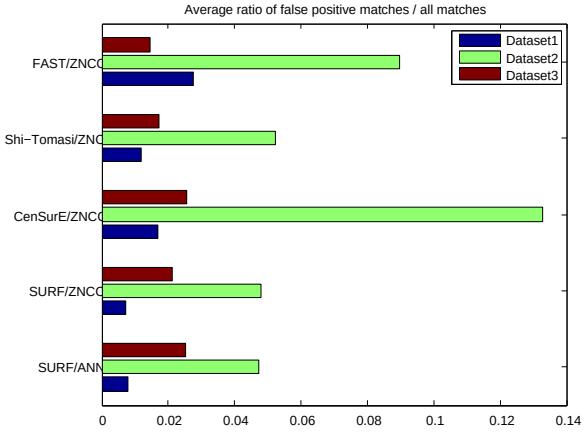


Figure 6. Average ratio of false positive matches / all matches. High value implies matching that is prone to outliers. SURF feature matching, and Shi-Tomasi features matched with ZNCC appear to be overall the most robust. All detectors did well on datasets 2 and 3, which were not blurry. FAST and CenSurE were more prone to outliers on Dataset2, which had the highest motion blur.

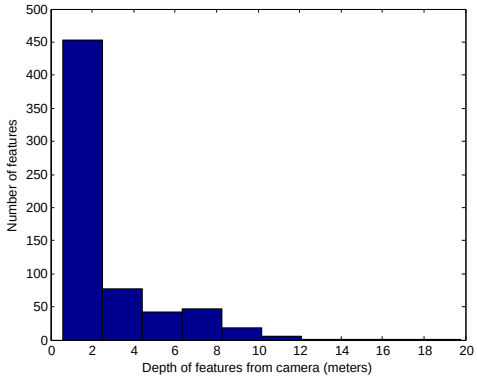


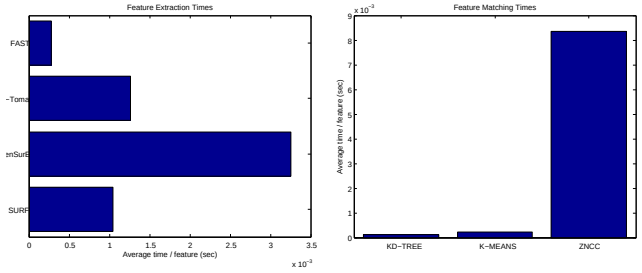
Figure 7. Distribution of estimated feature depths from the first camera frame in which they were seen.

D. 3D Feature Position Estimation

Offline processing of the ARToolKit tags that we placed underwater on the bottom of the experiment area, showed that the robot was swimming approximately 2.5 meters over the bottom, mostly parallel to it, so we will treat that as ground truth for the depth of each feature. After removing outlier paths using the epipolar constraints implied by Eq. (10), we rejected 3D feature position estimates that were outside the field of view of the robot, less than 0.5 meters or more than 10 meters away from the robot. The distribution of depths from the camera frame, for the remaining inliers is shown in Fig. 7, which suggests that the majority of inliers had depths that fell in the $[1, 4]$ meter range.

E. Running Time

We compared the average time spent on keypoint extraction and matching, normalized by the total number of keypoints processed. The results were obtained on an Intel Pentium Core 2 Duo at 1.6GHz and 4GB of RAM. The OpenCV C++ library implementations were used for each detector, descriptor and matching scheme. The results in Fig. 8(a) clearly show that SURF and the Shi-Tomasi detector spend approximately 1 millisecond on each feature, while CenSurE spends 3 milliseconds. The latter is most likely due to the fact that CenSurE computes keypoints at all scales. Fig. 8(b) depicts a very significant difference between the running times of Approximate Nearest Neighbor matching through either randomized kd-trees or K-means, and normalized-cross correlation matching. This is not surprising, as the latter has complexity $O(nm)$ where n, m are the number of features extracted from the previous and the current image, respectively. On the other hand, kd-trees are constructed in $O(nlogn)$ and queried in $O(logm)$. Therefore, if real-time processing of images is a requirement, limiting the number of features extracted by the detectors becomes necessary.



(a) (b)

Figure 8. (a) Average feature extraction times per feature. CenSurE is the slowest among the detectors that we evaluated. (b) Average feature matching times per feature. The normalized cross-correlation similarity score (ZNCC) is computed by a brute force comparison of all possible feature pairs, hence it requires more computation time than Approximate Nearest Neighbors. The time spent on computing the fundamental matrix is not included in these matching times.

F. State Estimation

Figure 9 presents two illustrative examples of the state estimation algorithm applied to *Dataset3*, in one case using the SURF detector adjusted so that it detects approximately 500-600 features per image, and in the other case using the the Shi-Tomasi detector, tuned so that it detects approximately 200-300 features. Less than half of those features

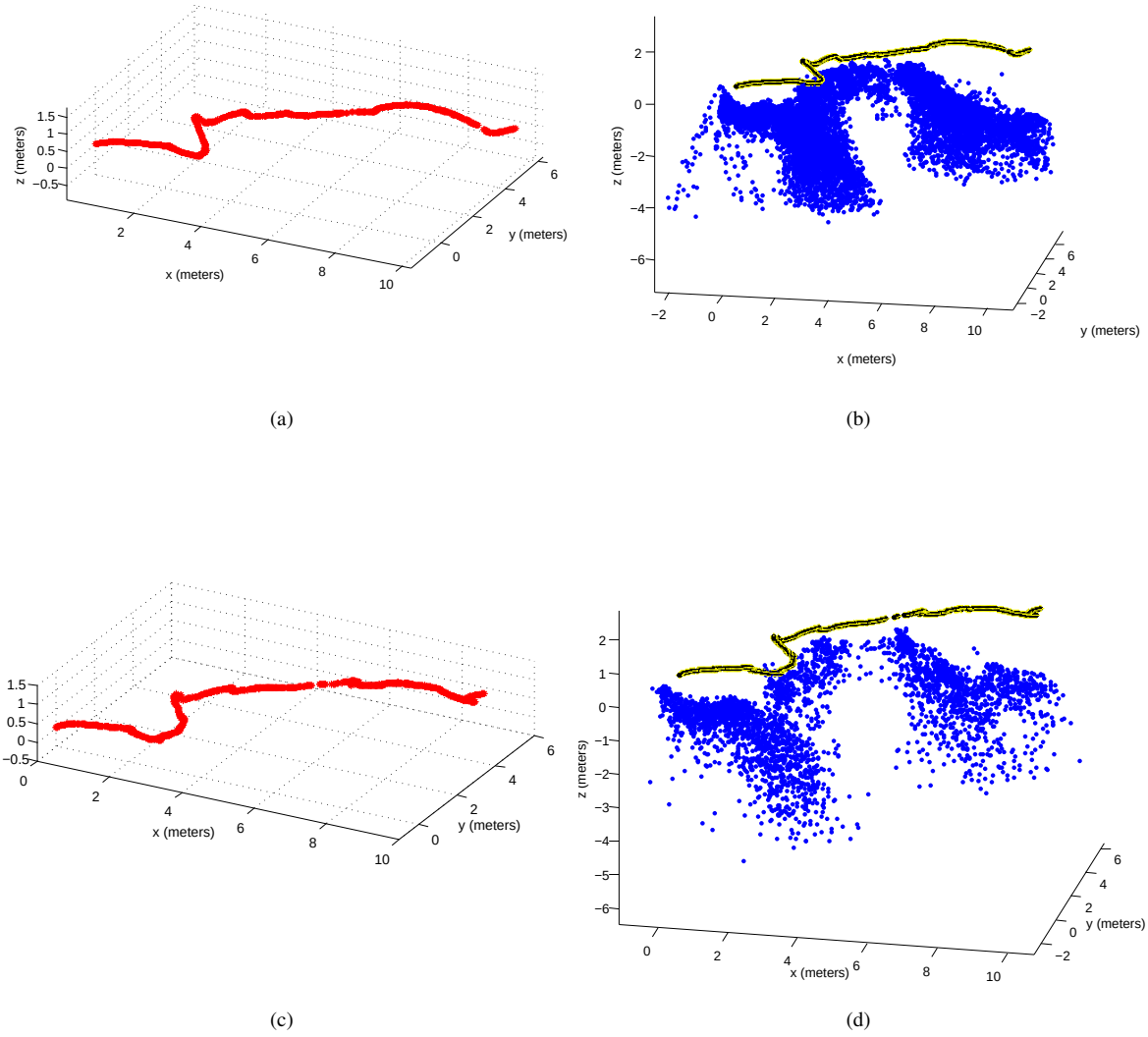


Figure 9. (a) Estimated trajectory for Dataset3, using SURF and Approximate Nearest Neighbor matching. (b) 3D coral structure, estimated via Eq. (8) along with the trajectory. (c) Estimated trajectory for Dataset3, using Shi-Tomasi features and ZNCC matching. (d) 3D coral structure

were matched from frame to frame, yet the robot's trajectory was reproduced at 1Hz camera frame rate for SURF and 3Hz frame rate for Shi-Tomasi. The estimated trajectory and 3D structure of the coral is shown in Fig. 9.

V. Conclusion

We performed an evaluation of combinations of different feature detectors, descriptors and matching strategies for the purposes of evaluating their suitability for our state estimation algorithm, which we want to use in underwater environments, in full 6DOF motion. We tested these combinations based on four criteria of interest: length of feature tracks, running time, false positive matches, and overall ability to facilitate estimation of 3D positions of feature from the camera frame. We used three underwater datasets that: (a) are representative of the environments that we want to run this algorithm in, (b) have sufficient lighting and motion blur variations, and (c) provided us with some notion of ground truth. The main conclusion of our evaluation is that the most suitable combination for real-time processing would involve either SURF keypoints matched via Approximate Nearest Neighbor search, or Shi-Tomasi features matched via ZNCC. On the other hand, the FAST and CenSurE detectors were shown to be inaccurate in image sequences with high levels

of motion blur.

ACKNOWLEDGMENTS

The authors would like to thank Philippe Giguere for assistance with robot navigation; Yogesh Girdhar and Chris Prahacs for their support as divers; Marinos Rekleitis and Rosemary Ferrie for their invaluable help with the experiments. The authors would also like to thank NSERC for its generous support.

REFERENCES

[1] J. Sattar, G. Dudek, O. Chiu, I. Rekleitis, P. Giguère, A. Mills, N. Plamondon, C. Prahacs, Y. Girdhar, M. Nahon, and J.-P. Lobos, "Enabling autonomous capabilities in underwater robotics," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, Nice, France, September 2008.

[2] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Rome, Italy, April 10-14 2007, pp. 3565–3572.

[3] P. Corke, C. Detweiler, M. Dunbabin, M. Hamilton, D. Rus, and I. Vasilescu, "Experiments with underwater robot localization and tracking," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2007, pp. 4556–4561.

[4] T. Barfoot, "Online visual motion estimation using fastslam with sift features," in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, 2005, pp. 579 – 585.

[5] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*. Acapulco, Mexico: IJCAI, 2003.

[6] R. A. Newcombe and A. J. Davison, "Live dense reconstruction with a single moving camera," in *CVPR*, 2010, pp. 1498–1505.

[7] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[8] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, 2005.

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

[10] J. Klippenstein and H. Zhang, "Quantitative evaluation of feature extractors for visual slam," in *Computer and Robot Vision, 2007. CRV '07. Fourth Canadian Conference on*, May 2007, pp. 157 –164.

[11] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework: Part 1," Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-02-16, July 2002.

[12] A. I. Mourikis and S. I. Roumeliotis, "A dual-layer estimator architecture for long-term localization," in *Proceedings of the Workshop on Visual Localization for Mobile Platforms*, Anchorage, AK, June 2008, pp. 1–8.

[13] N. Trawny and S. I. Roumeliotis, "Indirect kalman filter for 3d attitude estimation," University of Minnesota, Dept. of Comp. Sci. and Eng.,, Tech. Rep. 2005-002, March 2005.

[14] W. G. Breckenridge, "Quaternions proposed standard conventions," JPL, Tech. Rep. INTEROFFICE MEMORANDUM IOM 343-79-1199, 1999.

[15] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," in *ECCV*, 2006, pp. 404–417.

[16] J. Shi and C. Tomasi, "Good features to track," in *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, 1994, pp. 593 – 600.

[17] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European Conference on Computer Vision*, vol. 1, May 2006, pp. 430–443.

[18] M. Agrawal, K. Konolige, and M. R. Blas, "Censure: Center surround extremas for realtime feature detection and matching," in *ECCV (4)*, 2008, pp. 102–115.

[19] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *International Conference on Computer Vision Theory and Application VISSAPP'09)*. INSTICC Press, 2009, pp. 331–340.

[20] R. C. B. M. A. Fischler, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Comm. of the ACM,*, vol. 24, pp. 381–395, 1981.

[21] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004, pp. 245–246.

[22] [Online]. Available: http://www.hitl.washington.edu/artoolkit/