Air-SSLAM: A Visual Stereo Indoor SLAM for Aerial Quadrotors

Pompilio Araújo, Rodolfo Miranda, Diedre Carmo, Raul Alves, and Luciano Oliveira

***Abstract*— In this letter, we introduce a novel method for visual simultaneous localization and mapping (SLAM)—so-called Air-SSLAM—which exploits a stereo camera configuration. In contrast to monocular SLAM, scale definition and 3-D information are issues that can be more easily dealt with in stereo cameras. Air-SSLAM starts from computing keypoints and the correspondent descriptors over the pair of images, using good features-to-track and rotated-binary robust-independent elementary features, respectively. Then a map is created by matching each pair of right and left frames. The long-term map maintenance is continuously performed by analyzing the quality of each matching, as well as by inserting new keypoints into uncharted areas of the environment. Three main contributions can be highlighted in our method: 1) a novel method to match keypoints efficiently; 2) three quality indicators with the aim of speeding up the mapping process; and 3) map maintenance with uniform distribution performed by image zones. By using a drone equipped with a stereo camera, flying indoor, the translational average error with respect to a marked ground truth was computed, demonstrating promising results.**

***Index Terms*— Drone, stereo vision, visual simultaneous localization and mapping (SLAM).**

I. Introduction

VISUAL simultaneous localization and mapping (SLAM) can be treated as a problem of tracking geometric characteristics that appear naturally in an unknown environment. Image features are extracted, composing an environment map, which is constantly updated as the robot camera moves. To maintain this reference map, stored features are compared with those currently being captured by the camera. This process is continuously performed in order to provide robot's self-localization.

In monocular SLAM [1]–[9], initial camera moving is necessary to build the starting reference map. When using two cameras in a stereo configuration [10], position estimation can be done with no camera motion computation, since two views are available at the same time. Superimposing camera images are necessary to estimate the depth of objects, allowing each region of the space to be analyzed twice. That reduces the possibility that any noise or disturbance will systematically affect the system. One problem that occurs with monocular SLAMs is the lack of scale, which lately represents a multiplicative factor of the disparity in the function of two following image views. In a stereo configuration, scale is unique, and defined as the previously known distance between the two cameras.

Particularly for aerial quadrotor vehicles (drones), visual SLAM turns to be more complex due to the increase of one degree of freedom in the vertical direction. Very few works propose visual SLAM methods applied in drones [4], [10]. Engel *et al.* [4] present a system that allows for a drone to be controlled from a ground base computer; the parallel tracking and mapping (PTAM) method [2] is used to generate the environment map from the aerial vehicle; because PTAM was directly adapted from the original work, which was conceived for augmented reality, the SLAM method does not present the required response time to be used for aircraft control. Mur-Artal and Tardós [10] adapted their monocular system (ORB-SLAM) to stereo (ORB-SLAM2), solving the scale initialization problem, which increased the general performance of the system compared with the previous version. ORB-SLAM2 is an efficient stereo SLAM, which uses ORB [11] to work with the image keypoints; also, a place recognition is proposed in [10], based on bags of binary words (DBoW2) [12], a binary method that requires off-line training.

Nowadays, a drone equipped with cameras is essential to explore remote areas. Drones capable of automatically performing low-altitude flights in places, such as roofed archaeological parks, caves, or sites covered by dense vegetation, are of great importance to guarantee a certain level of monitoring in difficult-to-access areas [13], [14]. In this letter, we introduce a novel visual stereo-based SLAM for drones, so-called Air-SSLAM. Our method relies on good features-to-track (GFTT) [15] to extract keypoints, which are lately described by rotated-binary robust-independent elementary features (rBRIEF) [11], [16]. The keypoint descriptors are used to determine the environment map for a DJI Phantom 3 PRO drone. To estimate the drone's position, the keypoints of the two views are matched in order to define the transformation matrix between two consecutive timed views (in time t and $t + 1$). The drone's position is refined by an optimization method. Our main contributions are as follows.

1) A novel method of point matching, starting the search at a probable point location and then gradually increasing the search area. As each keypoint is found, the probable location for the subsequent points are updated and corrected.
2) Mapping is achieved by using three quality indicators to remove or include keypoints.
3) Instead of being based on keyframes as the existing works [4], [10], map maintenance is performed by

Manuscript received March 17, 2017; revised May 31, 2017 and July 7, 2017; accepted July 19, 2017. This work was supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico under Grant 447574/2016-0. *(Corresponding author: Luciano Oliveira.)*

The authors are with the Intelligent Vision Research Laboratory, Federal University of Bahia, Salvador 40170-110, Brazil (e-mail: lrebouca@ufba.br).

Color versions of one or more of the figures in this letter are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LGRS.2017.2730883

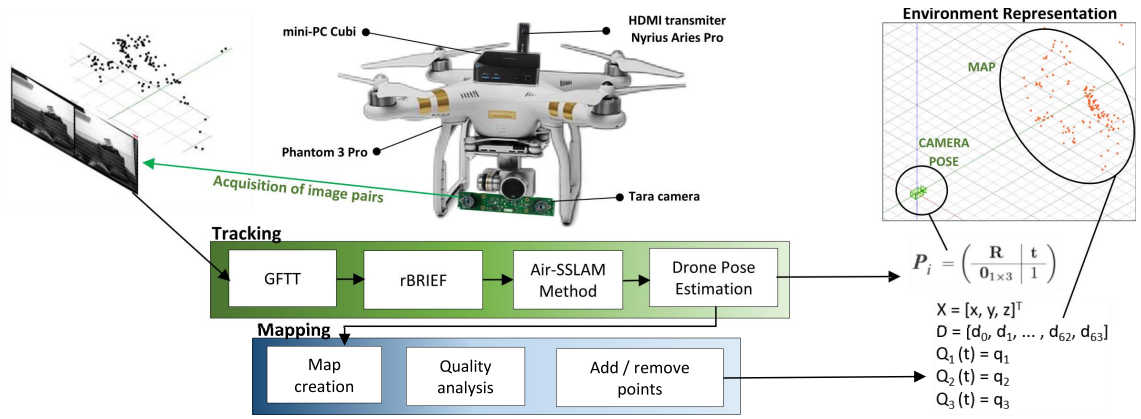

Fig. 1. Outline of Air-SSLAM: DJI Phantom 3 Pro is used as the testing drone. After stereo images are captured by the TARA cameras, keypoints are computed on the right and left images by using GFTT, in an MSI Cubi computer. A match between the right and left keypoints is performed over an rBRIEF descriptor calculated on the GFTT, and an initial map is generated with the depth estimation of each pair of keypoints. This process is subsequently repeated using the initial map as a reference. A Nyrius Aries Pro HDMI Transmitter is used to perform communication.

zones, so that the distribution of points is as uniform as possible, turning the process to be faster and with a better drone's pose estimation.

II. Outline of the Stereo Visual SLAM

Fig. 1 summarizes our proposed method depicting fivefold steps, which will be covered in detail in Sections III and IV, as follows.

1) A Tara stereo camera, comprised of two identical cameras, was placed with equal focal lenses, positioned side by side at a known distance (60 mm, in case), with parallel optical axes and planes. Image acquisition is simultaneously carried out, offering two frames with the dimensions of 640×480 pixels.
2) Detection of keypoints is performed by using GFTT [15], and made available in a frame structure along with the image; GFTT was specifically proposed to work with tracking, by treating cases of occlusion and noise that do not correspond to physical points in the world; GFTT is based on Newton–Raphson approximation to perform transformations between two similar images.
3) rBRIEF [11], [16] descriptor is used to represent the region around the keypoints, providing feature vectors, which are compared across the stereo images; rBRIEF is a binary-based feature descriptor, which runs faster than the scale-invariant feature transform (SIFT) [17] and speeded up robust features (SURF) [18], keeping good rotation invariance.
4) The results of 2) and 3) are provided to create an initial map, which is used as a reference to track the keypoints.
5) Mapping is performed on a different thread, and the quality of the available keypoints on the map is analyzed through of three indicators (keypoint detection rate, Hamming distance, and M-estimator error).

Software implementation of the method was performed by using three main separate tasks processed in parallel. The first one is responsible for capturing images, extracting characteristics of the environment, and processing frames. The second one performs descriptor extraction, feature tracking, and camera position estimation. The third task performs mapping, using the keypoint quality analysis. All these tasks are performed by considering an MSI Cubi computer with I5-5200u, 2.2GHz processor. Indeed, all experiments carried out here considered the MSI Cubi equipped in the drone. A Nyrius Aries Pro HDMI Transmitter was used to show image acquisition on-the-fly.

III. Representing the Environment

The two outputs of the system that represent the environment are camera pose and map.

A. Camera Pose

Let X_{Cj} denote the camera pose (position and orientation) in time j, being defined with six degrees of freedom $[x\, y\, z\, \phi\, \chi\, \psi]^T$, where x, y, and z are the coordinates of the camera position, and ϕ, χ, and ψ are the roll, pitch, and yaw rotation angles around camera reference coordinates, respectively. These six parameters define camera pose (position + orientation), and are mathematically represented by a 4×4 Euclidean special Lie group SE(3) matrix. The center of the left camera image plane is used as a reference for the coordinate system of the camera pose, and the right one is defined as $[(x + d)\, y\, z\, \phi\, \chi\, \psi]^T$, where d is the distance between the two cameras.

B. Map

A map is created using the first ten frames of each camera, by performing a matching process, which will be described in Section IV-C. Each point on the map (p_i) is defined with coordinates $X_{\mathrm{Ci}} = [x_i\, y_i\, z_i]^T$ in reference to the global world coordinate system. Subsequent frames are then processed, and new keypoints can be added as the environment is scanned, as well as be removed to save processing time. At the end, the map is comprised of a collection of geographically positioned 3-D points (keypoints), called here *map keypoints*. Besides point coordinates, map keypoints have a description

TABLE I
COMPARATIVE EVALUATION OF DETECTION TIME

Detection time (ms)				
SIFT	SURF	FAST	oFAST	GFTT
139.35	24.66	1.90	7.44	11.56

TABLE II
COMPARATIVE EVALUATION OF KEYPOINT DESCRIPTION TIME

Description time (ms)		
BRISK	FREAK	rBRIEF
17.88	11.76	11.06

information of the point and quality indicator, which will be used in the matching process; also an image zone information are included in the keypoint, as will be discussed in Section V.

IV. CAMERA SELF-LOCALIZATION

A. Detecting Keypoints

Table I summarizes the results of the processing time of some keypoint detection methods. A visual SLAM embedded in a drone must work in execution time, so that we need a feature extraction method that is able to run as fast as possible. This rules out the possibility of using SIFT (∼140 ms; octave layers = 3, contrast threshold = 0.4, edge threshold = 10, and $\sigma = 1.6$) or SURF (∼25 ms; Hessian threshold = 100 and octave layers = 3). Features from accelerated segment test (FAST) [19] (threshold = 10) also cannot be used as it does not include an orientation operator. Thus, we make tests using oFAST and GFTT. The choice of GFTT (maximum number of corner returned = 500, quality accept = 0.1, and minimum Euclidean distance = 3) over oriented FAST (oFAST) (threshold = 10) relied on the fact that the former qualitatively presented a more reasonable distribution of the keypoint detected in the image in our experiments.

B. Describing Keypoints

To perform the match, a description of the keypoints is necessary. The comparative analysis in Table II shows the time taken to describe 400 keypoints in a pair of frames. Three binary methods were used for evaluation: binary robust invariant scalable keypoints (BRISK) [20] (threshold = 30, num. octaves = 3, and pattern scale = 1.0), fast retina keypoint (FREAK) [21] (pattern scale = 22.0 and num. octaves = 4), and rBRIEF [11], [22]. rBRIEF (maximum number of features = 500, scale factor = 1.2, and edge threshold = 31) showed the best performance with respect to the time to compute the descriptor in one image. Another advantage of rBRIEF is the ability to add rotational and scale invariance without significantly increasing computational time.

C. Matching Keypoints

To match keypoints, it is necessary to estimate an *a priori* camera pose. For that, a decay velocity model is used, and

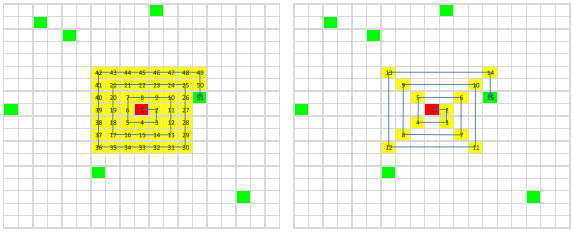

Fig. 2. In both images, each colored square represents an image pixel. Red: map keypoint. Green: current keypoints. Yellow: all pixels analyzed by the method before making a matching. (Left) Without the preparatory-search procedure. (Right) With the preparatory-search procedure.

defined as

$$P_{i+1} = \exp(0.9 \cdot \log(P_i - P_{i-1})) \cdot P_i \tag{1}$$

where P is the camera pose in time i. The decay rate was chosen to be 0.9, similar to the one found in [2]. It is noteworthy that camera pose refers to the center of the left camera plane, as defined in Section III-A. Using this *a priori* camera pose, a map keypoint, X_W, in world coordinate system, is translated to the camera coordinate system as X_C, according to

$$X_C = P^{-1} \cdot X_W. \tag{2}$$

In practice, when two new frames are acquired, new keypoints are detected over these frames. These initial keypoints are referred to *current keypoints*. Current keypoints are matched to the previously detected map keypoints. The very first goal with this task is to accurately calculate the camera pose. If a greedy method was chosen to match current keypoints to map keypoints, this task would take a long time, making the method infeasible to be used in on-the-fly applications. To decrease the matching time, a novel method that exploits information from the *a priori* camera pose is proposed in order to define the search area of the current keypoint. For each map keypoint, we look for the equivalent current keypoint within the previously defined search area. Within that area, the search is performed in a spiral path, ensuring that the closest current keypoints will have priority over the rest. To compare the similarity of the pair of keypoints (current and map), a Hamming distance between the keypoint descriptors in the pair is calculated. The pairs of keypoints that present values below a threshold ($dH_{\min} = 20$) are considered to be found.

That proposed search method demonstrated to improve performance when a reasonable estimation of the *a priori* camera pose is found, for example, when the camera is stopped or when the camera is moving slowly. In practical tests, the matching time quadratically increases, as the camera moves faster. To tackle the problem of poor estimation of an *a priori* camera pose, a preparatory-search procedure was conceived, such that, for each frame, vectors are generated to inform the position of each current keypoint. This allows for the spiral search not to go through all the pixels until it finds a current keypoint. Fig. 2 shows a map keypoint (red) being matched to a current keypoint (green) by using our search method; on the left figure, when there is no preparatory-search procedure, the spiral path goes through all pixels around the coordinate of the map point (yellow points); on the right figure,

as a preparatory-search procedure is implemented, the spiral path passes only through the corners and the most likely current keypoints. The algorithm complexity decreases from $O(N^2)$ to approximately $O(4N)$.

Another issue that significantly influences on the performance is the size of the search area. The search ends when the correspondent current keypoint is found with the Hamming distance less than $dH_{\min} = 10$; in some cases, the distance does not exceed that threshold, what turns the method to go to an endless loop. At this point, since it is not possible to know about the quality of the *a priori* camera pose, the initial search area begins very large. As the search proceeds, the size of the search area decreases according to the geometric distance of the current keypoints found. A large search area is defined for the first points (10 initial points), and then the geometric distances for these points are calculated, and the window size is adjusted according to the measured values.

D. Accurately Calculating the Camera Pose

Given a set of pairs of correspondent keypoints (map and current), now an as accurate as possible camera pose is necessary. Two cases can be found here: 1) if there is no *a priori* camera pose (or this pose is unreliable), and then a random sample consensus is used; this procedure is iteratively repeated N times in order to reduce the error or 2) when there is a good estimation of the camera pose (evaluated after the keypoint matching procedure with the geometric distance); on having an initial solution for the camera pose, P_{3D}, we are able to apply an M-estimator optimization algorithm, similar to that presented in [2], but modified to the case of 3-D points, as follows:

$$P_{3D} = \arg\min_{P} \sum_{j \in S} f(|e_j|, \sigma_T) \tag{3}$$

where σ_T is a parameter of Tukey's biweighted objective function, and e_j, corresponding to the jth keypoint, is the projection error, given by

$$e_j = X_{\mathrm{Nj}} - P \cdot X_{\mathrm{Mj}} \tag{4}$$

where $X_{\mathrm{Nj}} = [x\ y\ z]^T$ and $X_{\mathrm{Mj}} = [x\ y\ z]^T$ denote current and map keypoints, respectively. After that, we apply a Kalman filter to stabilize the calculated camera pose, using the following parameters: number of $states = 18$, measure states $= 6$, and time of measurement $= 0.525$ ms.

V. Mapping

Mapping procedure relies on the creation of the initial map, and subsequent map maintenance. As mentioned before, a map is indeed a set of keypoints. As it is not possible to maintain all keypoints detected in the images, quality indicators were conceived to add or remove keypoints to keep the map as compact as possible without loosing relevant information. The first indicator is the *keypoint detection rate*, which represents the detection rate considering the estimation of the last 30 keypoints (the higher this value, the higher is the indicator). The second indicator is the *Hamming distance*, and represents the degree of similarity between the current and map keypoints (the farther, the worser this indicator). The third indicator is the **M**-*estimator error*, which is related to camera pose estimation; in the M-Estimator algorithm the e_j error of each point j is used to calculate the indicator, so that the smaller the error, the greater is the indicator value.

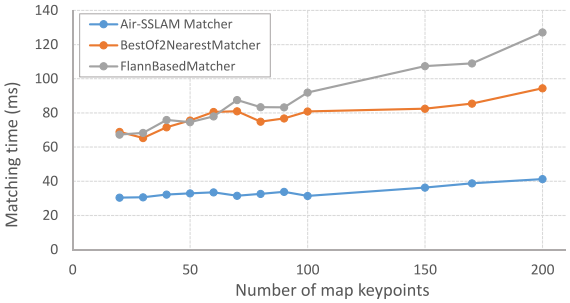

Fig. 3. Comparison of Air-SSLAM with two other methods: BestOf2NearestMatcher and FlannBasedMatcher.

Map maintenance is accomplished by image regions, referred here as zones. Zones with several map keypoints are classified as rich zones, while zones with few keypoints are classified as poor. Rich and poor zones are found by an adaptive threshold considering the M-estimator error. New current keypoints (green dots), located in poor areas, can be added as map keypoints, if the quality indicators are at an expected level.

A. Bundle Adjustment

An optimization procedure based on Levenberg–Marquardt algorithm is used to locate new keypoints as they are inserted. This procedure is necessary to minimize the error between the current (new) keypoint and the map (already inserted) keypoint. All map keypoints and the camera pose calculated are used to perform this refinement, also known as bundle adjustment, as that found in [23].

VI. Experimental Analysis

Two experimental evaluations were carried out to assess the performance of Air-SSLAM: the processing time of our keypoint matching and the translational average error (TAE) considering our SLAM system in a drone. We comparatively evaluate the processing time with two other methods implemented in OpenCV library: BestOf2NearestMatcher and FlannBasedMatcher. The result is summarized in Fig. 3. The matching was accomplished by increasing the number of map keypoints. With 200 map keypoints, our method takes 40 ms to run, while BestOf2NearestMatcher runs almost twice slower, and FlannBasedMatcher runs more than three times slower. Fig. 1 shows the equipment used to run the experiments. Drone flights were monitored by two independent external cameras, evaluating the TAE with respect to known wall markers. Fig. 4 (Left) shows the variation of the three components of the position (in meters), during a test flight, compared with the ground truth (black line); on the right, the figure shows the drone trajectory estimated by the system. Average errors of 0.10, 0.20, and 0.20 m were found in the x-, y-, and z-directions, respectively. It is noteworthy that the trajectory without Kalman filter turns the drone movement unstable (gray

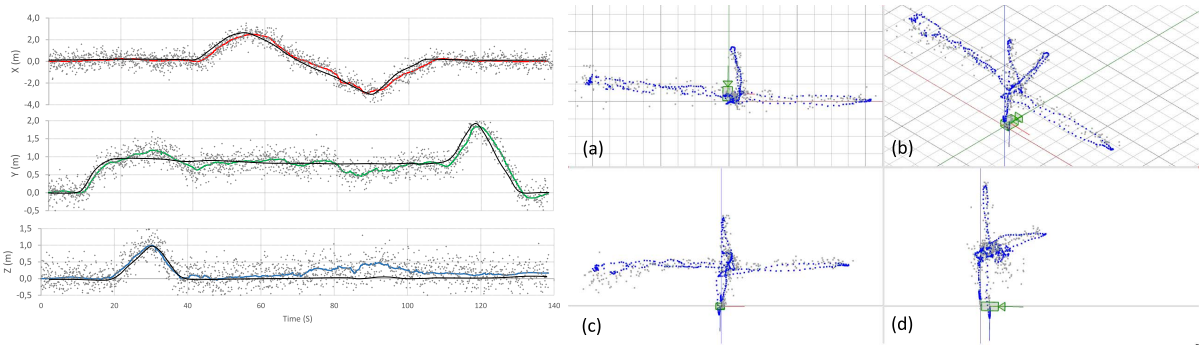

Fig. 4. Average translational error. (Left) Three coordinates of the camera pose during a drone flight; ground truth is in black. (Right) Trajectory of the drone during the tests (gray dots represent trajectory without Kalman filter; red, green, and blue are with Kalman filter). (a) Top view. (b) 3-D view. (c) Rear view. (d) Lateral view.

dots in the left figure). To assess the depth error associated with the stereo configuration, an experiment was carried out by measuring the distance between an object with a marker and the stereo image plane. The distance was found with our stereo system and with a high precision laser tape measuring. After that, the error between the two distances (measured and calculated) was computed. We found errors of 10 cm at 1 m, 25 cm at 3 m, and 49 cm at 5 m. As it was expected, the furthest the distance from the object, the greater the error of stereo depth. With this error, we recalculated the position of the image keypoint to adjust the precision.

VII. Conclusion

A stereo visual SLAM, called Air-SSLAM, applied on drones was presented. Rather than being based on keyframes, Air-SSLAM directly performs a search for keypoints in an efficient way. Keypoint matching is performed using an *a priori* camera pose in a spiral-based search. Map maintenance takes place by continuously analyzing three quality indicators and image zones, making this procedure to be performed with keypoint uniform distribution. That turned the method to run almost in a quasi-constant time when using less than 200 keypoints. We are working now on a loop-close method to increase mapping accuracy in large environments. In the future, we can use the zone approach to detect the keypoints by adaptively adjusting the detection thresholds. This way, we can improve the distribution of keypoints in space, regulating the processing time of SLAM method.

References

[1] D. R. dos Santos, M. A. Basso, K. Khoshelham, E. de Oliveira, N. L. Pavan, and G. Vosselman, "Mapping indoor spaces by adaptive coarse-to-fine registration of RGB-D data," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 2, pp. 262–266, Feb. 2016.

[2] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. IEEE/ACM Int. Symp. Mixed Augmented Reality (ISMAR)*, Nov. 2007, pp. 225–234.

[3] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, Jun. 2007.

[4] J. Engel, J. Sturm, and D. Cremers, "Camera-based navigation of a low-cost quadrocopter," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 2815–2821.

[5] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, Nov. 2011, pp. 2320–2327.

[6] M. Michael, T. Sebastian, K. Daphne, and W. Ben, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2003, pp. 1–6.

[7] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2003, pp. 1403–1410.

[8] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.

[9] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 834–849.

[10] R. Mur-Artal and J. D. Tardós. (Oct. 2016). "ORB-SLAM2: An open-source SLAM system for monocular, stereo and RGB-D cameras." [Online]. Available: https://arxiv.org/abs/1610.06475

[11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, 2011, pp. 2564–2571.

[12] D. Gálvez-López and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1188–1197, Oct. 2012.

[13] D. Meyer, E. Lo, A. Hoff, M. Hess, and F. Kuester, "Utility of low-cost drones to generate 3D models of archaeological sites from multisensor data," in *Proc. Soc. Amer. Archaeol. (SAA)*, 2015, pp. 30–35.

[14] S. Wechsler, C. Lipo, C. Lee, and T. L. Hunt, "Technology in the skies: Benefits of commercial UAs for archaeological applications," in *Proc. Soc. Amer. Archaeol. (SAA)*, 2016, pp. 36–42.

[15] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 1994, pp. 593–600.

[16] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary robust independent elementary features," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2010, pp. 778–792.

[17] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[18] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2006, pp. 404–417.

[19] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2006, pp. 430–443.

[20] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary Robust invariant scalable keypoints," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Nov. 2011, pp. 2548–2555.

[21] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast retina keypoint," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, pp. 510–517.

[22] E. Karami, S. Prasad and M. Shehata, "Image matching using sif, surf, brief and orb: Performance comparison for distorted images," in *Proc. Newfoundland Electr. Comput. Eng. Conf.*, St. John's, NL, Canada, Nov. 2015.

[23] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G^2o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2011, pp. 3607–3613.