Performance of monocular and stereo camera in indoor environment for Visual SLAM using ORB method

Tianshu Ruan[1], V.Amrusha Aryasomyajula[1] and Nasser Houshangi[1]
[1]Department of Electrical and Computer Engineering, Purdue University Northwest, Indiana, USA
rtszxxz@gmail.com, amrushaarys@gmail.com, nhousha@pnw.edu

2022 IEEE International Conference on Electro Information Technology (eIT) | 978-1-6654-8009-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/eIT53891.2022.9813775

***Abstract*—Visual Simultaneous Localization and Mapping (vSLAM) has received much research interest for real time applications like autonomous navigation. This paper presents the performance of vSLAM between monocular and stereo camera systems for a mobile robot in indoor environment using ORB-SLAM2 method. The approach performance is experimentally validated between the two cameras by using a Jackal mobile robot from Clearpath Robotics.**

Keywords—vSLAM, ORB, Visual Odometry, oFAST, rBRIEF.

I. INTRODUCTION

Visual Simultaneous Localization and Mapping (vSLAM) refers to the problem of using images as the external information to establish the position of the robot and build a map of the environment. When the robot is in an unknown environment, a positioning component is required to solve the self- localizing problem [1]. Sensors used for self-localizing can be classified in two categories. The first category is non-intrusive sensors like: wheel encoders, lasers, cameras, lidar, etc. The second category is intrusive sensors which are present in the environment for example: bar codes [2], landmarks, stickers, etc. The main limitation with intrusive sensor is that the localization system can function properly and provide a general solution only if they meet all the constraints even though they are simple and reliable. In contrary the non-intrusive sensors such as lasers, cameras, etc. can observe physical quantities rather than direct locations. The advantage of using these sensors is that they can obtain very precise and dense information of the environment. Therefore, the use of non-intrusive sensors to solve and obtain SLAM is the focus. With vSLAM, the camera is used to solve the positioning and map construction problem. The main reason to use cameras to perform vSLAM is that they can obtain information like appearance, color and texture of the environment which allows the robot to perform high-level tasks like recognition of places and people. Cameras can be divided into categories like: Monocular, Stereo and RGB-D. A monocular camera has only one lens, stereo has two or more lens and RGB-D camera has capability of collecting color images and providing depth information from depth sensor. Oriented FAST and rotated BRIEF (ORB)-SLAM [3] is one of the vSLAM solutions compatible for all three types of cameras as implemented in this research.

II. METHODOLOGY

ORB-SLAM is a feature-point-based algorithm to collect feature pairs between two images. The process of visual odometry is shown in Fig. 1. The oriented Features from Accelerated Segment Test (oFAST) algorithm is used as an improved feature detector based on FAST algorithm [4]. The rotated Binary Independent Robust Elementary Features (rBIREF) algorithm is the feature descriptor which helps to find feature point pairs. Then, based on camera used in the ORB-SLAM system, the methods are different to restore the camera trajectory and camera poses. Monocular camera implements epipolar constraint and triangulation while stereo and RGB-D camera uses Iterative Closest Point (ICP) and Perspective-n-Point Camera Pose Estimation (PnP). This paper focuses on implementing the novel oFAST and rBRIEF method.

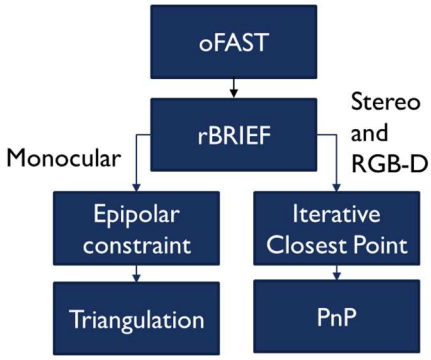


Figure 1. ORB visual odometry structure.

Feature detector is based on FAST algorithm. FAST corner point is defined as: If a pixel has a large difference between enough pixels among its surrounding neighborhood, the pixel may be a corner point. The steps for the FAST algorithm are described as follows:

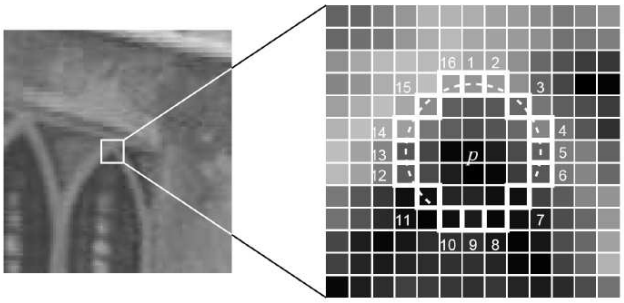


Figure 2. neighboring pixels are considered in the FAST algorithm. Figure source:[4]

1. As shown in the Fig. 2, a circle with a radius of 3 pixels entered on the pixel p has 16 pixels (p1, p2, ..., p16).
2. A threshold is defined as 30% of the pixel p's gray level. The gray level difference between p1 and center p is calculated. The same calculation can be done between p9 and the center p. If their absolute values are less than the threshold, p point cannot be a feature point and skip; otherwise, as a candidate point, it needs further evaluation;
3. If p is a candidate point, the pixel difference between p1, p9, p5, p13 and center p is calculated. If at least 3 of their absolute values exceed the threshold, the point remains as a candidate point; otherwise, it is skipped;
4. If p is a candidate point, the pixel difference between p1 to p16 and center p is calculated. If at least 9 of their absolute values exceed the threshold, the point remains as a candidate point; otherwise, it is skipped;
5. Calculate the FAST score value of the feature point which is the sum of the absolute values of the 16 points to the center difference, that is, the s value, and filter a neighborhood (such as 3x3 or 5x5) centered on the feature point p. If there are multiple feature points, determine the s value of each feature point. The point with the largest s value is saved, and the rest can be skipped. If there is only one feature point (corner point) in the neighborhood, it needs to be saved.

ORB algorithm makes some modifications to improve the FAST algorithm by adding the orientation to each feature point. The modified algorithm is called Oriented FAST. The feature points extracted by original FAST algorithm does not contain the parameter for the direction. Later, in [3] a method is proposed to find the directions which provided a solution to solving the rotation problem in feature point matching part.

The ORB algorithm proposes the use of moments to determine the direction of the FAST feature points. That is to say, the centroid in the range of radius r of the feature point is calculated by the moment [5]. The feature point coordinates to the centroid form a vector indicating the direction of the feature point. The moment is defined as follows:

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y) \tag{1}$$

where I(x, y) is the image grey level expression and p,q are 1,0 and 0,1. The first order moment m_{10} and m_{01} is used to describe the centroid of the image C as:

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}\right) \tag{2}$$

where zero order moment m_{00} is the sum of the gray levels of each pixel in the range of r radius of the feature points. Then, the angle θ between the feature point and the centroid is defined as the direction of the FAST feature point:

$$\theta = 2(m_{01}, m_{10}) \tag{3}$$

FAST feature points have no scale invariance. It means that if the scale of the same image change, the feature points extracted by FAST are different for the two images. A scale pyramid shown in Fig. 3 could solve this problem.

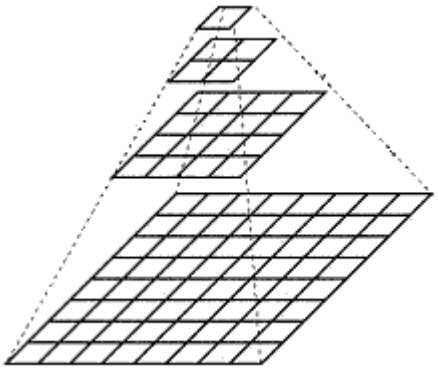


Figure 3. Image pyramid with different layers.

The original image lays at the bottom of the pyramid. As the pyramid moves up to the top, both size and resolution will be reduced. In the ORB algorithm, the scale factor of the pyramid is 1.2 which means the ratio of the pixel area from close layers is 1.44. The number of pyramid layers is set to 8 layers. In other words, including the original image, a total of 8 different sizes of images were generated. The grey level of scaled image I' is expressed as:

$$I' = \frac{I}{(ScaleFactor)^k} \tag{4}$$

where k represents the number of image level and I is the original image grey level. The sums of the feature points extracted from 8 different scales of image are regarded as the FAST feature points.

For a better matching quality between two images, an improvement on traditional BRIEF algorithm is proposed called rBRIEF [3]. It is assumed that the original BRIEF algorithm [6] selects n pairs of points D in the neighborhood pixels of the feature point SxS (generally S = 31).

$$D = \begin{pmatrix} x_1, x_2, \ldots, x_{2n} \\ y_1, y_2, \ldots, y_{2n} \end{pmatrix} \tag{5}$$

Rotate matrix D by the angle θ to get a new point pair:

$$D_\theta = R_\theta D \tag{6}$$

The method mentioned in the original BRIEF algorithm to select point pairs was abandoned. Instead, another method to

reselect point pairs collection (rBRIEF) is proposed [16]. For each point in the data set, its 31x31 pixels neighborhood is considered. Gaussian smoothing is applied to the image which means the average gray level in a 5x5 neighborhood of a point is used, instead the actual grayscale value of the point. This feature improves its ability to resist noise. There are (31-5) x (31-5) =676 such sub-windows for neighborhood of 31x31 pixels. After eliminating the overlap pairs, all the possible pairs are M=205590. The following steps to select 256 pairs from the data set is used:

1. Assume that the total number of feature points is 300k, a matrix Q with 300k rows and M columns is built. Each column represents the binary code according to the results from rBRIEF;
2. An average is calculated for each column in Q matrix. According to the distance from the average value to 0.5, resort the column vector from small to large and generate a new matrix T;
3. Select the first column in T as the vector R;
4. Calculate the correlation between the next column in T (1x300k) and all the vectors in R(1x300k) separately. If the absolute correlation is smaller than the threshold, the column vector in T is moved to R;
5. Repeat step 4 until there are 256 vectors in matrix R. If there are less than 256 vectors in matrix R, the threshold in step 4 is increased.

The rBRIEF combines the rotation invariance and low correlations together. Therefore, it is chosen to select feature points pair in ORB-SLAM. If the paired feature points are obtained, the camera pose and map points location can be calculated based on these data.

After the encoding process, a 256-bit binary code is obtained for each feature point in the image. It is possible to match the two images with similar or overlapping parts. Hamming distance [7] is used to determine feature point matching. Two rules are applied in this part:

1. If the Hamming distance of the feature corresponding to two pixels is greater than 128, the two points must not be paired.
2. If the feature corresponding to two pixels has the smallest Hamming distance, it is paired.

After the feature extraction and matching tasks are completed, the camera's motion based on the matched point pairs is estimated. The approach taken depends on the camera used which is briefly mentioned in Fig. 1.

For a monocular camera, only the 2D pixel coordinates are known, so the problem is to estimate camera motion based on two sets of 2D points. This problem is solved using the epipolar geometry [8]. Then, the position of feature points is restored by triangulation. For stereo, RGB-D cameras, the camera motion is estimated based on two sets of 3D points. This problem is solved by Iterative Closest Point (ICP). Then, the position of the feature points is restored by "Perspective-n-Point Camera Pose Estimation" (PnP) [9].

The ORB-SLAM system can receive images from stereo, monocular and RGB-D cameras [10], and turn the original image into a frame containing all the information needed for the ORB-SLAM to perform vSLAM. It includes camera intrinsic matrix, distortion parameters, timestamp, scale pyramid information, feature descriptor, depth information, bag of words, camera pose, rotation matrix, translation vector, map points before and after correction, and reference keyframe.

ORB-SLAM system is divided into tracking, local mapping, and loop closing as shown in Fig. 4.

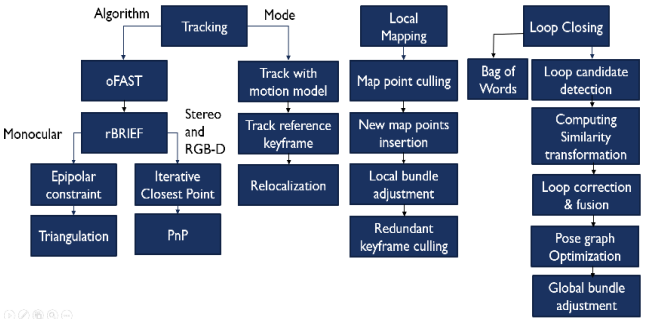


Figure 4. Framework of ORB system.

A. Tracking

The main task of tracking thread is to locate the camera with each frame using Visual Odometry (VO) and decide whether to insert or remove a new keyframe. The tracking thread includes three models: the motion model "Track with Motion Model", the keyframe "Track Reference Keyframe" and the "Relocalization". The three tracking models are designed to obtain a rough initial value of the camera pose, and then Bundle Adjustment is performed for further pose optimization.

1) Track with Motion Model

The motion model solves the estimated pose based on the constraint relationship between the two frames. Assuming that the camera is moving at a constant speed, which means R and t is the same with the previous motion. The paired feature points are calculated. If there are enough matched points, this model tracks successful. This motion model is suitable for situations where the speed and direction of motion are relatively consistent and no big rotation involved, such as cars, mobile robots and etc. If there are not enough paired feature points, this model will fail and the "Track Reference Keyframe" model will be applied as described next.

2) Track Reference Keyframe

If the first motion model fails, the last keyframe is used to restore the camera motion, as the distance between the current frame and the previous keyframe is not very big. The bag of words is used to speed up matching and bundle adjustment (BA) is used to correct the pose. The procedure is shown below:

1. Calculate the bag of words (BoW) of the current frame [11][12].

2. Compare the current BoW with the previous keyframe BoW. If the matched words more than 20, the BA is used to correct the pose.

3) Relocalization

If the matching between the current frame and the nearest neighbor keyframe also fails, it means that the current frame has been lost and the real position cannot be determined. At this point, the only way to find the location is to match all the keyframe using BoW as the procedure is shown below:

1. The BoW is found according to the current keyframe.
2. Detect the candidate keyframes based on the similar BoW.
3. Filter out the keyframes with the numbers of BoW whose range are in 0.8-1 times maximum BoW matched.
4. Calculate the similarity score for the BoW matching quality.
5. Calculate the cumulative similarity score for keyframes which have the same observation area.
6. Find out the keyframe with the highest BoW score plus same observation score.
7. Utilize PnP to estimate pose and relocalization.

B. Local Mapping

The main task of mapping thread is to receive and process new keyframes, check for new map points, generate new map points, implement local bundle adjustment, maintain the accuracy of local maps and control the quality and scale of keyframe sets. It has four modules to keep the map accurate and in a small size which are Map Point Culling, New Map Points Insertion, Local Bundle Adjustment and Redundant Keyframe Culling.

C. Loop Closing

The Loop Closing thread is a very important part of the SLAM system. Due to the cumulative error (drifting) of the VO process, the main task of the loop closing thread is to detect the revisited places using Bag of Words (BoW) [13], that is, to determine whether the location was visited before. After detecting, the similarity matrix Sim3 is calculated to close the loop. At last, the global Bundle Adjustment [14] is applied to optimize the keyframe poses and map points to keep cumulative error to an acceptable range.

III. EXPERIMENTS AND RESULTS

Before performing experiments, camera calibration is a prerequisite for accurate feature detection. All the intrinsic and extrinsic parameters of monocular and stereo cameras are calculated through calibration. The calibration method is Zhang's camera calibration algorithm [15] which is used to estimate the camera lens parameters and to understand the size and distortion of the features extracted. This calibration method is a plane-based self-calibration method which uses views of planar calibration pattern, whose metric dimensions are known.

A. Experimental Platform

The experimental platform used Jackal mobile robot as shown in Fig. 5.

Figure 5. Jaclal mobile robot with Kinect and stereo cameras.

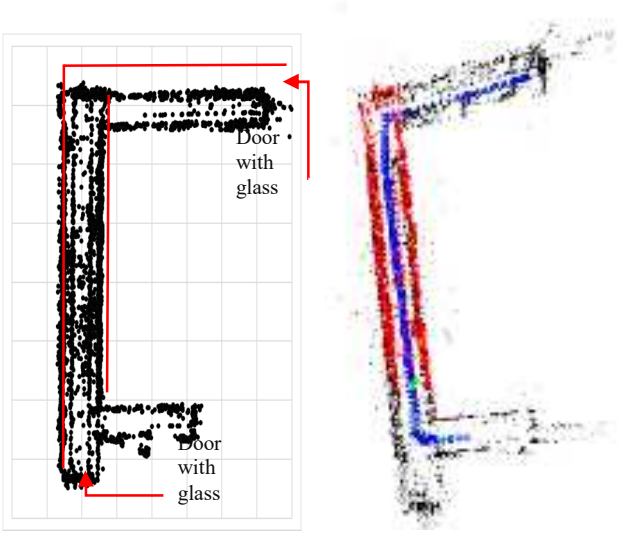
The Jackal robot's motherboard carries a 2.4GHz Celeron J1800 dual core processer with 2GB memory card. Ethernet and WIFI are both available for the communication. Ubuntu 14.04 is installed as the operating system. The ROS Indigo is included in Ubuntu operating system and is used to manipulate the robot. An isolated network is built to connect the host computer to the robot. The Bumblebee2 stereo camera is mounted on the front deck of the robot. The camera connects to the standard IEEE1394b port on the robot's motherboard and provides 640 × 480 resolution color images at 48 Frames Per Second (FPS) from two Sony ICX204 sensors. This camera is capable of running, in both monocular and stereo camera mode. The other camera used is Kinect V1 Red Green Blue- Depth (RGB-D) camera that is a combination of a monocular camera and an infrared sensor.

The experiments are performed in two different indoor environments by constructing the map using Kinect, monocular and stereo cameras. In this work, the complete trajectory of the indoor environment obtained from the RGB-D camera is considered as ground truth trajectory due to high accuracy.

B. Performance in A Small-scale Environment

In this experiment, the actual environment used is an indoor corridor. The interior of this environment includes doors, walls, lights, notice boards etc. The 2D map constructed using RGB-D camera is shown in Fig. 6(a). This map is filtered by a pass-through filter which is commonly used to remove the sparse points in point cloud maps. The redline reveals the walls in this environment. There are some map points beyond the red line at the top right. The reason of this phenomenon is that there is a door with glass at the end of this corridor. The camera takes features behind the doors via glass which leads to some points laying outside the corridor. The camera also extracts features from the ceiling and the ground, which leads

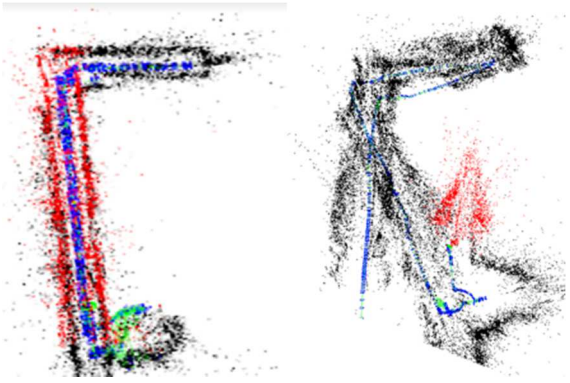
to many map points laying in the middle of the corridors in the 2D map.



(a) (b)

Figure 6. (a) 2D map from the RGB-D Camera. (b) 3D map and camera trajectory obtained by RGB-D camera.

In the 3D map, the blue squares represent keyframes, the black points represent the optimized map points. The red points represent the map points extracted from the current frame. The green square shows the current camera location with respect to the environment. The trajectory obtained by RGB-D camera is shown in Fig. 6(b). The trajectories of corridor environment obtained using monocular and stereo cameras is shown in Fig. 7.



(a) (b)

Figure 7. (a) Camera trajectory obtained by monocular camera. (b) Camera trajectory obtained by stereo camera.

From the maps obtained by monocular and stereo cameras, the monocular camera map is similar to the ground truth whereas there is a noticeable variation in the map obtained by the stereo camera due to high computational complexity during stereo matching.

To compare the accumulated errors between the trajectories between monocular and stereo SLAM, Root Mean Square Error (RMSE), mean, standard deviation is calculated as shown in Table I.

TABLE I. CAMERA TRAJECTORY ERROR BASED ON THE RGB-D RESULTS FROM THE CORRIDOR

	Monocular	**Stereo**
RMSE(m)	0.693	1.952
Mean(m)	0.565	1.770
Standard Deviation(m)	0.402	0.822
Min(m)	0.122	0.285
Max(m)	1.823	3.063

From map and the table, it's easy to notice that the monocular SLAM provides lower deviations and shows better performance than stereo SLAM for our platform. The stereo SLAM loses tracking several times during the scan which leads to the map from stereo camera misplaced. RMSE is calculated to indicate the performance of the trajectories obtained from monocular and stereo camera compared to RGB-D camera.

C. Performance in A Computer Lab Environment

In this experiment, the actual test environment used is a typical letter "E" shape indoor computer lab environment. The environment interior consists of tables, chairs, desktops, electronic equipment, cables, projector etc. The trajectory of the complete environment obtained from Kinect camera is considered as reference or ground truth which is shown in Fig. 8. The red points represent the map points extracted from the current frame. The green square shows the current camera location according to the environment.

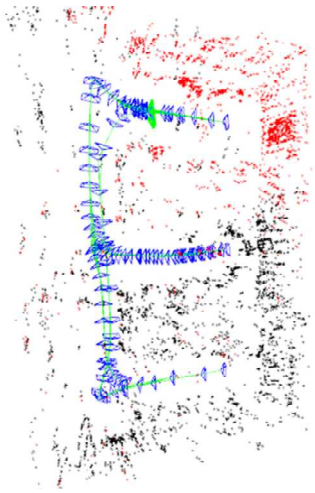
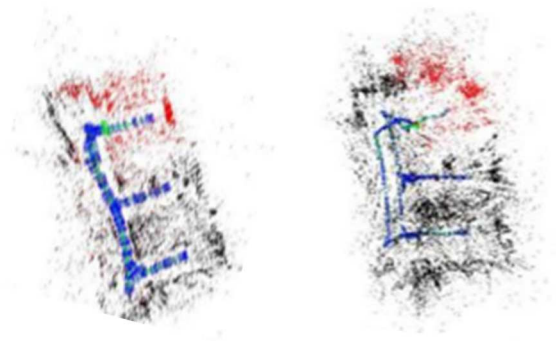


Figure 8. Camera trajectory obtained by RGB-D camera.

The trajectories of computer lab obtained from monocular and stereo cameras is shown in Fig. 9. The comparison between monocular and stereo camera performance is made by calculating RMSE.



(a) (b)

Figure 9. (a) Trajectory obtained from monocular camera. (b) Trajectory obtained from stereo camera.

From the maps obtained by monocular and stereo camera, the monocular camera map is similar to the ground truth whereas there is a noticeable variation along the length connecting three aisles in the map obtained by stereo camera due to high computational complexity during stereo matching.

TABLE II. CAMERA TRAJECTORY ERROR BASED ON THE RGB-D RESULTS FROM THE LAB

	Monocular	**Stereo**
RMSE(m)	0.599	0.668
Mean(m)	0.532	0.588
Standard Deviation(m)	0.274	0.316
Min(m)	0.117	0.133
Max(m)	1.257	1.229

The map of monocular SLAM is close to the RGB-D map. The RMSE also indicates that monocular SLAM performs better in the lab with the current experimental platform. Considering the laboratory is a smaller environment than corridor, the stereo SLAM does not lose tracking during motion, the RMSE values are close between monocular and stereo results.

IV. CONCLUSIONS

Stereo camera has more computational complexity compared to monocular as to perform stereo matching of the images found in both cameras. Hence, a faster processor is required to implement the algorithm effectively. Also, for stereo cameras, the quality of output and disparity will greatly depend on the scene. As for a white wall, tabletops, desks and etc. with fewer textures returns very few feature point matches while performing stereo matching. Hence, the performance of monocular becomes better than stereo. If the computation capability gets upgraded, the performance of the stereo SLAM is expected to be better than the monocular SLAM. But the feature points matching problem still remains when facing feature-lack scenarios.

REFERENCES

[1] Tim Bailey and Hugh Durrant-Whyte. Simultaneous localization and mapping: Part 2. *IEEE Robotics and Automation Magazine*, Pages 108-117, September 2006.

[2] Jorge Fuentes-Pacheo, Jose Ruiz-Ascencio, Juan Manuel Rendon-Mancha. "Visual simultaneous localization and mapping: a survey". *Springer Science & Business Media Dordrecht*, (2015) 43:55-81, Pages 55-81, November 2012.

[3] Raul Mur-Artal, J.M.M.Montiel, Juan D. Tardos. "ORB-SLAM: A versatile and accurate Monocular SLAM system*". IEEE Transactions on robotics*, Vol.31, No.5, Pages 1147-1163, October 2015.

[4] Edward Rosten, Reid Porter and Tom Drummond, "FASTER and better: A machine learning approach to corner detection". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010, pp 105–119.

[5] M. K. Hu, "Visual Pattern Recognition by Moment Invariants", *IRE Trans. Info. Theory*, Vol. IT-8, 1962, pp.179–187.

[6] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary Robust Independent Elementary Features," *European Conference on Computer Vision (ECCV)*, Hersonissos, Greece, Pages 778–792, September 2010.

[7] Hamming, R. W. "Error Detecting and Error Correcting Codes." *Bell System Technical Journal*, Vol. 29, No. 2, 1950, pp. 147–160.

[8] Hartley, Richard, and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. 2004, doi:10.1017/cbo9780511811685.

[9] Lepetit, Vincent, et al, "EPnP: An Accurate O(n) Solution to the PnP Problem." *International Journal of Computer Vision*, Vol. 81, No. 2, 2008, pp. 155–166.

[10] Raul Mur-Artul and Juan D. Tardos. "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras". *IEEE Transactions on robotics*, Vol.33, No.5, Pages 1255-1262, October 2017.

[11] Galvez-Lopez, Dorian, and Juan D. Tardos. "Real-Time Loop Detection with Bags of Binary Words." *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[12] Jutta Willamowski, Damian Arregui, Gabriella Csurka, Christopher R. Dance, Lixin Fan, "Categorizing nine visual classes using local appearance descriptors." *Icpr Workshop on Learning for Adaptable Visual Systems (2004)*.

[13] D. G´alvez-L´opez and J. D. Tard´os, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, Pages 1188–1197, 2012.

[14] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon, "Bundle Adjustment — A Modern Synthesis" Vision Algorithms: *Theory and Practice Lecture Notes in Computer Science*, 2000, pp. 298–372.

[15] Zhengyou Zhang, "A Flexible New Technique for Camera Calibration", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Pages 1330 – 1334.

[16] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," 2011 International Conference on Computer Vision, 2011, pp. 2564-2571.