Visual odometry based relocalization using ORB feature descriptor.

Animesh Nema
Advisor - Dr.Michael A.Gennert
Department of Robotics Engineering
Worcester Polytechnic Institute

***Abstract*— The goal of this paper is to develop a computationally inexpensive relocalization system for SLAM (simultaneous localization and mapping) through feature-based techniques. Binary feature descriptor techniques such as ORB(Oriented FAST and Rotated BRIEF) can be used to detect key points and find similarities between two images, in real-time and with very less computational cost. Furthermore, the partial illumination, scale and rotational invariance of ORB makes it slightly more robust and ideal for this purpose. Once, a suitable match has been found, geometric transformation between current image and the image from map can be used to estimate a relative pose of the robot. This can be helpful in enabling the robot to quickly relocalize itself in case of tracking failures or when initializing the robot in the same environment, later. The approach was tested on TUM RGB-D SLAM benchmark data set and requires only monocular images for feature matching and pose estimation. Depth images are required only to compute the final part, i.e., the distance of the camera from the scene.**

I. INTRODUCTION

In the past decade, vast amounts of research has been done on visual SLAM such as LSD-SLAM, ORB SLAM[1], ORB SLAM2[2] etc. This can be credited to the availability of cheap and powerful cameras such as Kinect, that are capable of extracting rich information from the environment, in both, monocular and stereo format. As a result, there have been advents in the work of SLAM with researchers using stereo-imaging, 3D point clouds etc. Also, significant contributions in the field of computer vision have been made over the same period and have been open sourced for the greater mass to learn, use and build upon it.

An often discussed problem in SLAM is that of relocalization. Relocalization is an important aspect in any SLAM system, where the robot estimates it's position and orientation, using prior knowledge about the environment(map). In case of visual SLAM, the environment is mainly mapped using the data from the camera. If for some reason, the robot loses track of it's position in the environment, the camera feed can be used to help the robot find it's location in the environment.

A lot of papers have been written on complete SLAM systems and loop-closure techniques. However, there have been very few papers focused on the aspect of relocalization even though it's an integral part of SLAM. The motivation of this project is to develop a computationally inexpensive relocalization module that can quickly generate an estimate of the robot's location in the environment by just utilizing the camera feed.

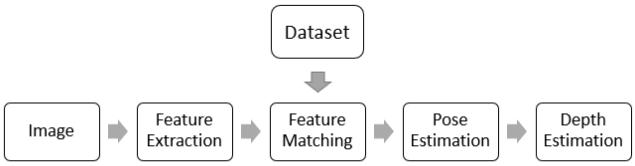

Fig. 1. Pipeline of the relocalization system

As seen in figure 1, the system can be broken down into three main components. First, ORB is used to extract key points from a scene and then store them in the form of binary feature vectors. These binary feature vectors are then compared with the ones already present in the data base.This is called as feature matching. As soon as a suitable match is found, the corresponding image is retrieved from the map.

Once the image has been retrieved, the robot has some idea of it's position in the environment. However, it is still unaware of it's orientation. The angle between current image and corresponding image can be determined using geometric transformation, hence, giving the relative pose of the camera.

The third aspect is to determine the distance of the camera from the scene. This can be done by utilizing the depth data from the camera. The depth image generated by the kinect will provide information about how far the camera is from the scene by reading the value the pixels.

In this way, the robot can estimate it's position and orientation in the map fairly quickly and with very little computational load, using just a camera. The process is discussed in detail in sections III, IV and V.

II. LITERATURE REVIEW

In [1] the authors introduce a novel approach called ORB SLAM, a feature-based monocular SLAM system for both indoor and outdoor environments. The system is robust to severe motion clutter, allows wide baseline loop closing and relocalization. The highlight of the paper is how ORB can be used as an alternative to SIFT or SURF, for the purpose of place recognition.

The authors present an improved version of ORB SLAM in [2]. In addition to monocular images, ORB SLAM2 works on stereo images as well. A lightweight localization mode is also presented that leverages visual odometry tracks for unmapped regions and gives a zero-drift localization.

In [3] the authors present a detailed comparison between different gradient histogram-based feature descriptors such as SIFT or SURF and binary feature descriptors such as BRIEF, BRISK, ORB etc. The authors test these descriptors on two different publicly available data sets and compare their performance based on time taken, accuracy etc. This paper gives a good insight to the accuracy vs time complexity trade off, while also highlighting advantages and disadvantages of each of the techniques.

A comparison of loop closing techniques has been presented in [4]. The authors explain different types of ways in which correspondences can be generated, namely, image-to-image, image-to-map and map-to-map. The paper also discusses the use of RANSAC to eliminate the outliers and improve accuracy.

In [5] the authors present an approach for efficiently recognizing and identifying previously observed areas. The paper talks about identifying and extracting key points from a scene, in order to create a database and evaluate the performance of knn (k-nearest neighbour) algorithm applied to key point voting based solution for place recognition.

The authors in [6] describe a biologically inspired approach called Rat SLAM, based on computational models of rodent hippo campus for Visual SLAM ground based platforms. The authors use just a monocular camera to accurately perform loop closures and relocalization through sequences of familiar visual scenes.

The work in [7] focuses specifically on relocalization using binary features, as a means of a recovery strategy. The authors use Locality sensitive hashing(LSH) to boost the feature matching process. They use 3-point pose method and Progressive Sample Consensus(PROSAC) for pose estimation. Finally Levenberg Marquardt optimization (M-Estimator) is applied to refine the pose.

III. Oriented FAST and Rotated BRIEF

Fig. 2. Key points detected by ORB on an image.

ORB, developed by OpenCV, is a binary feature descriptor algorithm that uses a combination of FAST(Features from Accelerated Segments Test) and BRIEF(Binary Robust Independent Elementary Features) algorithms.

ORB is widely used for feature extraction purposes owing to the fact that it is computationally inexpensive as compared to previous techniques such as SIFT and SURF while also maintaining a comparable performance.

ORB is also scale and rotation invariant. It can also deal with illumination and noise variations up to a certain extent. In other words, it's quite a robust algorithm with an acceptable trade-off between accuracy and time complexity. As a result, it's a good candidate for a real-time application, on a device with less computational resources.

A. FAST

FAST is a feature detector that can quickly detect areas of interest in the image. For each pixel 'p', it compares it with 16 surrounding pixels that fall within a small circle, as seen in figure 3.

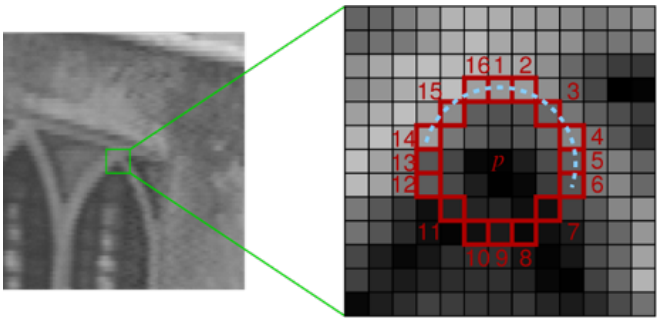

Fig. 3. Working of FAST feature detector.

If 8 or more pixels are over or under a certain threshold value i.e., if they are deemed brighter or darker than the pixel 'p', the pixel is selected as a key point. FAST identifies numerous key points in the image based on user input and stores the location of these key points. The location of the key points will be used later in the project to account for pose estimation.

ORB executes FAST, multiple times for each image, by scaling the image down by a pre-determined factor. This makes ORB, scale invariant.

B. BRIEF

BRIEF converts the extracted key points into binary feature vectors comprising of 1's and 0's.

$$V = [101000001111...]$$

The way BRIEF works is that it first smoothens the image by applying a gaussian blur. This makes ORB slightly less susceptible to noise. BRIEF then selects a random pair of pixels around a key point. The first pixel is selected from a gaussian distribution with a standard deviation, say 's', centered around the key point. The second pixel is selected in the same manner, but, with a standard deviation 's/2' centered around the first pixel.

BRIEF then starts constructing the feature descriptor. If the first pixel is brighter than the second pixel, it assigns the value of 1 to the corresponding bit, else it assigns it a value of 0.

Since computers deal with binary codes, it is easier for a computer to process such information, which is why it is ideal for a real-time application.

ORB uses R-BRIEF, which can detect a key point in different orientations, thus making ORB, rotationally invariant.

IV. Dataset

I have used the publicly available TUM RGB-D SLAM Benchmark dataset[8] for relocalization. The dataset contains RGB images along with corresponding depth images taken from a kinect.

Fig. 4. RGB Image from the TUM data set.

The data also contains the following groundtruth values:-

- Timestamp of each image.
- Translation values(tx ty tz) that give the position of the optical center of the camera with respect to the world origin as defined by the motion capture system.
- Orientation values (qx qy qz qw) in the form of unit quaternions that give the orientation of the optical center of the camera with respect to the world origin as defined by the motion capture system.

I made a few modifications in the data set, in order to test my relocalization module. Firstly, I took out a few images from the data set in order to use them as test images or 'current image'. These images were used as if the camera is seeing them for the first time while performing relocalization.

Secondly, I executed ORB on the remaining RGB images of the data set and stored the resulting binary feature vectors in a text file, in the form of an array. In other words, it can be thought of as an extension of the ground truth data. These binary vectors are a part of the ground truth values corresponding to each image.

V. Approach

Once, a map was created, which in this case is the sequentially stored images in the data set along with the ground truth values. Relocalization was performed by feeding a random image of the environment which was not part of the map. The pipeline of the relocalization module is described below:-

A. Feature Matching

ORB was performed on the given test image which will be referred to as the 'current image'. The binary feature vectors generated were then compared with the already stored vectors in the data set.

The algorithm is designed to proceed only if a suitable match is found. i.e., if the number of matches are less than a certain threshold (say 65%), then the program returns "No match found". The threshold value can depend on the type of data set and is user specific. In case of a data set with very similar scenes and features, the threshold value can be set slightly higher to reduce number of suitable matches.

Once a suitable match is found(the image frame with highest key points matched), there are 2 images to proceed with (the 'current image' and the best-match.)

B. Pose Estimation

Pose Estimation can be done in two ways.

1) Quaternion multiplication: The TUM data set consists of the orientation of the optical center of the camera in the form of quaternions.

For each set of frames obtained after feature matching, quaternion multiplication can be performed to reveal the angle difference of the camera between both images.

$$q1 = [x1, y1, z1, w1]$$

$$q2 = [x2, y2, z2, w2]$$

$$q3 = q1 * conj(q2)$$

$$\theta = 2 * arccos(w3)$$

However, in an actual scenario, the quaternion values with respect to the world frame will be either unavailable for the 'current image' or will be computationally expensive to generate.

Therefore, for the sake of keeping the algorithm quick, low-cost and vision based only, I also used geometric transformation between images.

2) Geometric Transformation: Geometric transformation is a widely used concept for various applications such as geographical mapping, removal of geometric distortion etc. I have used geometric transformation to estimate the relative pose between 2 images.

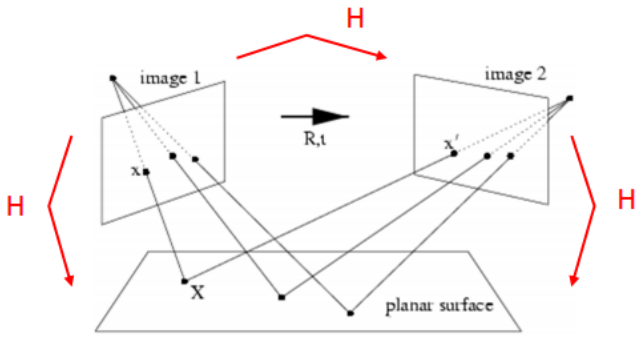

Fig. 5. Depiction of Homography.

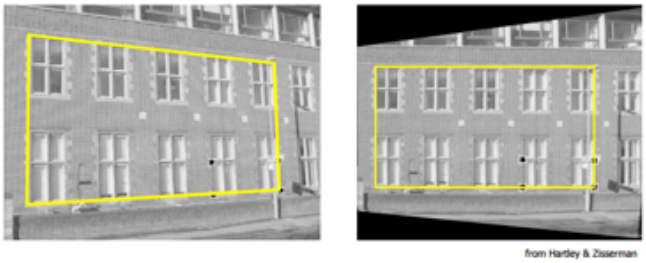

Fig. 6. Visualizing perspective transform.

Homography can be thought of as mapping of key points from one image on to another image. By finding the same key point in 2 images at different locations and mapping it, a perspective transform can be obtained. As seen in the figure 5and 6, it can help in determining the relative pose of the camera between 2 images.

This is where the key point location obtained by FAST is useful. I used it to track the different locations of the same key point in both images in order to carry out a perspective transform. RANSAC was used to minimize the outliers and improve the accuracy.

A 3x3 transformation matrix was obtained with the help of which, the relative pose was established between 2 images.

C. Depth Estimate

The depth estimation is the final part of this relocalization module. Thus far, only monocular RGB images were used to extract information. Once, a pose estimate is obtained, the depth image from kinect is used to generate an estimate of the distance.

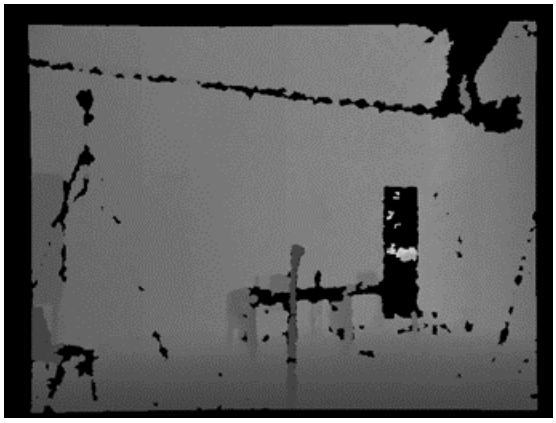

Fig. 7. Depth image from Kinect.

The camera's intrinsic parameters play a crucial role for depth estimation. The focal lengths, optical center etc are different for each camera. In the TUM data set, the depth images have been scaled by a factor of 5000. i.e., pixel value of 5000 corresponds to a difference of 1 metre.

By reading the pixel values, an estimated distance is achieved for the current location of the camera. Thus completing the relocalization module.

VI. Results

The relocalization module was tested on the TUM data set and the following results were obtained.

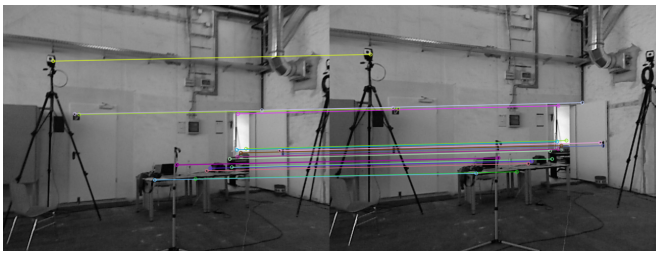

Fig. 8. Visualization of feature matching.

As seen in the figure 8, the image on the left is extracted from the data set after it is selected as the best match for the image on the right(current image). The image displays 20 best matching features between both images.

Multiple test runs were performed and the process took an approximate time of 0.60 seconds to find the best match among 1000 candidates.

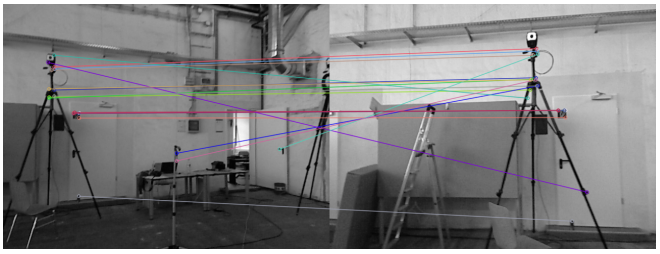

Fig. 9. Visualization of false positives.

There were some false positives as well as seen in figure 9. These were due to the fact that those key points were quite close enough in terms of how they are processed by ORB. However, the false positives were minimized by tweaking certain parameters such as scale factor, threshold for appending or rejecting matches etc.

These false positives are also not a major concern for 2 reasons. Firstly, as long as the map is rich, i.e., there are a lot of images in quick succession throughout the map, images with false positives will not qualify for the best suitable match, since, any image that is closer to the 'current image' will generate more number of key point matches with little to no false positives.

Secondly, when the image reaches the pose estimation part, key points are filtered based on the distance metric and also, RANSAC is used to eliminate the outliers. This will minimize the risk of proceeding with the false positives.

Regardless, in a feature friendly environment, ORB performed quite well giving accurate results for most cases. It was seen that the performance was better on images which were feature rich than those where it was hard to detect any features, for example white plain walls.

VII. Conclusion and Future work

- Based on the observations, ORB can be used to quickly extract key points in a scenario and accurately find a match. The process is computationally inexpensive and great for devices with less computational resources.
- It is a good candidate for relocalization due to it's robustness and ability to perform in real-time.
- However, it relies on a feature friendly environment and it's performance will degrade quickly in an environment where it is difficult to extract salient key points.
- Geometric transformation can be employed to vision-only based systems to perform pose estimation given the angle difference isn't too large between the images.

Hence, for an optimum performance, a rich map is required where a lot of images are stored in quick succession, hence reducing the chances of large angle differences.

In the future, I would like to work on the following aspects:-

- Explore the effect of relationships between key points in an image, on the performance of the system.
- Further expand the work to develop a low cost SLAM system with minimal sensors.

References

[1] Mur-Artal, R., Montiel, J. M., Tardos, J. D. (2015). ORB-SLAM: A Versatile and Accurate Monocular SLAM System. IEEE Transactions on Robotics,31(5), 1147-1163. doi:10.1109/tro.2015.2463671

[2] Mur-Artal, R., Tardos, J. D. (2017). ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. IEEE Transactions on Robotics, 33(5), 1255-1262. doi:10.1109/tro.2017.2705103

[3] Hartmann, J., Klussendorff, J. H., Maehle, E. (2013). A comparison of feature descriptors for visual SLAM. 2013 European Conference on Mobile Robots. doi:10.1109/ecmr.2013.6698820

[4] B. Williams, et al., A comparison of loop closing techniques in monocular SLAM, Robotics and Autonomous Systems (2009). doi:10.1016/j.robot.2009.06.010

[5] Zlot, R., Bosse, M. (2009). Place Recognition Using Keypoint Similarities in 2D Lidar Maps. Experimental Robotics Springer Tracts in Advanced Robotics,363-372. doi:10.1007/978-3-642-00196-342

[6] Milford, M., Wyeth, G., Prasser, D. (2004). RatSLAM: A hippocampal model for simultaneous localization and mapping. IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA 04. 2004. doi:10.1109/robot.2004.1307183

[7] Straub, Julian Hilsenbeck, Sebastian Schroth, Georg Huitl, Robert Mller, Andreas Steinbach, Eckehard. (2013). Fast Relocalization For Visual Odometry Using Binary Features. 2013 IEEE International Conference on Image Processing, ICIP 2013 - Proceedings. 10.1109/ICIP.2013.6738525.

[8] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, A bench-mark for the evaluation of RGB-D SLAM systems, in Proc. IEEE/RSJ Int.Conf. Intell. Robots Syst., Vilamoura, Portugal, Oct. 2012, pp. 573580.