*UDC 621.391*

# METHODS OF EXTRACT FEATURES FROM IMAGE IN VISUAL SIMULTANEOUS LOCALIZATION AND MAPPING

CHEN YIMING

*Belarusian State University of Informatics and Radioelectronics, Republic of Belarus*

*Received April 11, 2024*

**Abstract.** Visual simultaneous localization and mapping (*VSLAM*) depends on features from image to decide if two frame is relevant and reconstruct the map. This paper reviews multiple methods for extract features from images in grayscale, *RGB* and infrared.

*Keywords:* image feature, visual *SLAM*.

## Introduction

Simultaneous Localization and Mapping (*SLAM*) finds utility across a diverse spectrum of applications, including aerial and underwater mobile robots, autonomous vehicles, drones, and physical gaming environments.

In recent years, with the widespread use of high-quality, high-resolution cameras, the cost of computer processing images has become increasingly high. Visual *SLAM* algorithms rely heavily on image processing and analysis, making it crucial to extract image features and use them to reduce the computational cost of subsequent processing. As shown in Figure 1, the flowchart describes the basic steps of a general *SLAM* system.

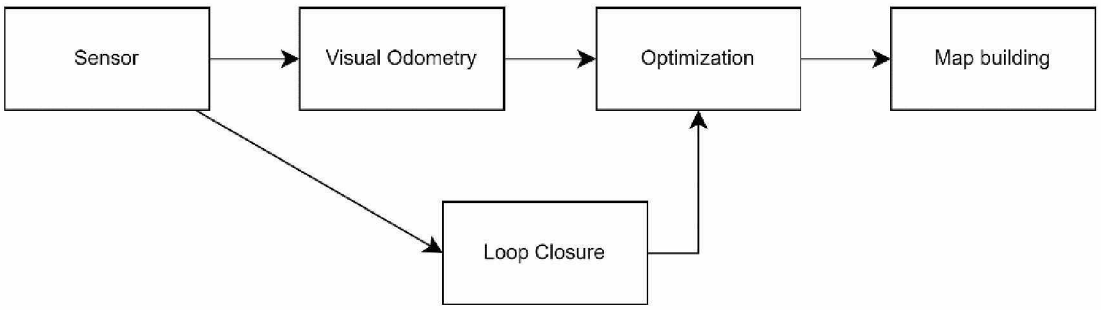


Figure 1. General SLAM flowchart

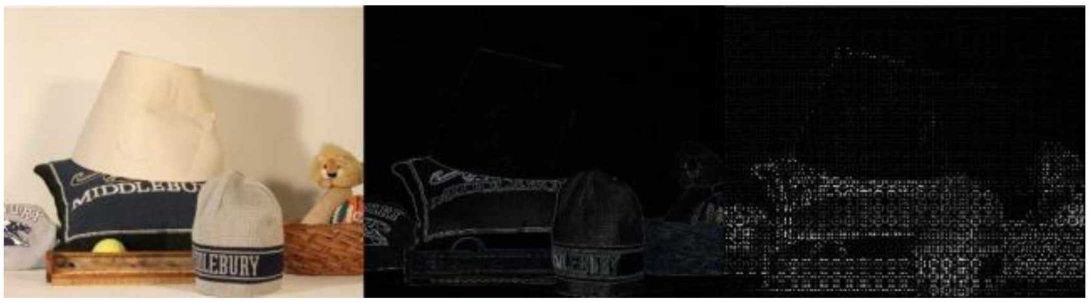
Visual odometry and loop closure depend on the features of the image. Visual odometry uses features from the image to speed up the computation of camera rotation and translation in the world. Loop closure, on the other hand, relies on database of features to determine when two sets of features indicate that the camera is returning to where it was, and then uses this information to correct the cumulative error of the system.

Features in an image refer to pixels that share common attributes and are different from other nearby pixels. Vision based *SLAM* detects the visual features of the environment such as corners, edges, colors, shadows, shapes, and depths. Features used for visual *SLAM*, must have rotation, orientation, translation, scaling, and luminance invariance.

## Methods to Extract Feature

To find the best features in an image that are direction, angle and contrast invariant is important to achieve higher accuracy and faster results in navigation and *SLAM*. Several modern techniques have been proposed for key point feature extraction and matching. Some of the best examples are Scale Invariant Feature Transform (*SIFT*), Speeded Up Robust Features (*SURF*), Features from Accelerated Segment Test (*FAST*), Binary Robust Independent Elementary Features (*BRIEF*), Oriented FAST and Rotated BRIEF (*ORB*) and Histogram of Oriented Gradients (*HOG*).

Dalal and Triggs proposed the basic model of Histogram of Oriented Gradients (*HOG*) for human recognition in 2005 [1]. In this feature description algorithm, the main feature is the distribution (histogram) of gradient (oriented gradient) directions. Using the image gradients, the corners and edges of the object can be extracted and defined as the main object features. This algorithm is one of the basic algorithms before deep learning techniques are introduced for feature description. Figure 2 shows the magnitude of gradient for an image and *HOG* result.



*a* *b* *c*

Figure 2. HOG feature detection algorithm [2]: *a* – original image; *b* – the magnitude of gradient; *c* – HOG result

Lowe proposed an image rotation and scaling invariant algorithm called *SIFT* in 2004 [3]. This technique is widely used for keypoint feature matching and is robust to image scaling and rotation, while being partially invariant to illumination changes and affine transformations. The algorithm utilizes a Taylor expansion of a Difference of Gaussian (*DOG*) scale space function and then adjusts it to center at a candidate point, thus achieving precise keypoint localization. Figure 3 shows an example of keypoints in two images matching each other using *SIFT* algorithm.

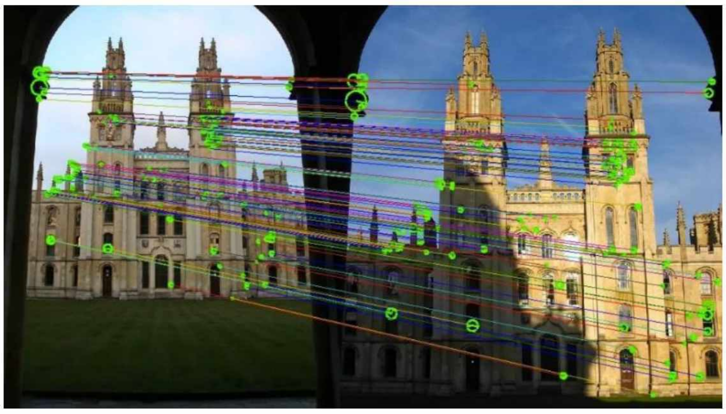


Figure 3. SIFT algorithm: Feature Matching in two images of the same building

*SURF* is a fast version of *SIFT*, proposed by Bay, Ess, Tuytelaars and Van Gool in 2008 [4], which generates multilevel image and descriptor pairs. The *SURF* algorithm works like *SIFT*, but it uses Haar wavelets instead of Gaussian differences. The *SURF* algorithm uses Hessian matrix determinants to select the positions and scales to generate fast and accurate descriptors.

The *ORB* (Orientation *FAST* and Rotation *BRIEF*) algorithm is an alternative descriptor proposed by Rublee et al. in 2011 [5]. The method is based on the *BRIEF* descriptor and the *FAST* keypoint detector, which measures the orientation of an angle by its intensity centroid. The moments of an image patch can be expressed as

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

where $q$ and $p$ represent the order of the moment, $x$ and $y$ are the spatial coordinates within the image.

With these moments, the center of mass of the patch can be defined as

$$C = (\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}). \quad (2)$$

Using $C$, the vector from the corner's center, $O$ is computed as $OC$. The orientation of the patch is defined as

$$R = \operatorname{atan2}(m_{01}, m_{10}), \quad (3)$$

where atan2 is the quadrant-aware version of arctan.

Using $R$, it is possible to rotate it to a canonical rotation to compute the descriptor, thus obtaining some rotation invariance. The *ORB* algorithm is very fast but less effective in terms of scale. This approach is used in the *ORB-SLAM* method proposed by Mur-Artal et al. in 2015 [6] and in the *ORB-SLAM* second version method proposed by Mur-Artal & Tardós in 2017 [7], which is a feature-based *SLAM* technique that uses an *ORB* feature descriptor to generate graphs from the keyframes. Figure 4 shows the *ORB* image matching results for a sample image.

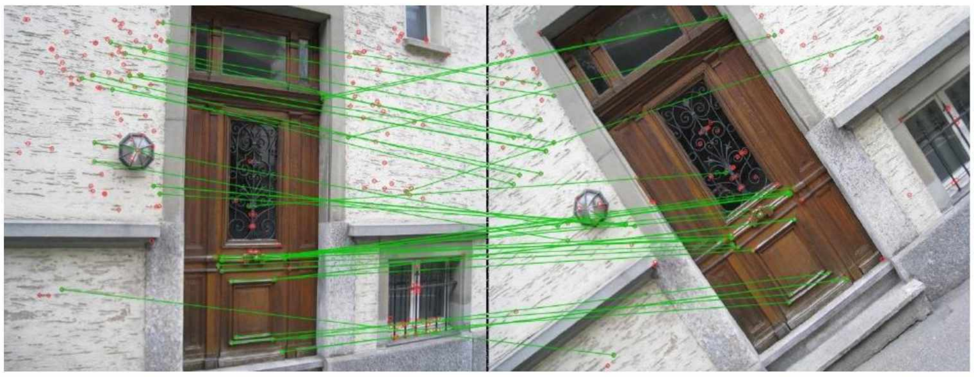


Figure 4. ORB algorithm: Image matching from original image and rotated original image [8]

## Conclusion

*SIFT*, *SURF* and *ORB* are commonly used feature extraction and description algorithms in image processing and target recognition. The main differences between them are computational complexity, scale and rotation invariance, feature description method, and number and stability of features.

In terms of computational complexity, *SIFT* has higher computational complexity in the feature extraction and matching phases, while *SURF* is relatively faster but still requires a lot of computational resources. *ORB* outperforms *SIFT* and *SURF* in terms of computational speed because it combines a fast corner detector and a rotated *BRIEF* descriptor.

As of scale and rotation invariance, *SIFT* has better invariance to scale changes and rotation transformations and can detect feature points at different scales and angles, *SURF* also has some scale and rotation invariance, but it is slightly inferior to *SIFT* and *ORB* lacks a little in this aspect, especially when dealing with large scale changes and rotations.

## References

1. Dalal, Navneet, Triggs, Bill. // 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). 2005. Vol. 1. P. 1–8.
2. Kazerouni, Iman Abaspur, et al. // Expert Systems with Applications. 2022. Vol. 205. P. 117734.
3. Lowe, David G. // International Journal of Computer Vision. 2004. Vol. 60. P. 91–110.
4. Bay, Herbert, et al. // Computer Vision and Image Understanding. 2008. Vol. 110(3). P. 346–359.
5. Rublee, Ethan, et al. // 2011 International Conference on Computer Vision. 2011. P. 2564–2571
6. Mur-Artal, Raul, Martinez Montiel, Jose Maria, Tardos, Juan D. // IEEE Transactions on Robotics. 2015. Vol. 31(5). P. 1147–1163.
7. Mur-Artal, Raul, Tardós, Juan D. // IEEE Transactions on Robotics. 2017. Vol. 33(5). P. 1255–1262.
8. Leutenegger, Stefan, Chli, Margarita, Siegwart, Roland Y. // 2011 International Conference on Computer Vision. 2011. P. 2548–2555