# Comparison of Outlier Filtering Methods in Terms of Their Influence on Pose Estimation Quality

Mohammed Hammoud, Melaku Getahun, Sergey Lupin

***Abstract*— Local feature matching is an important problem in many applications of computer vision. Matching the descriptors depending only on the distances is not enough since normal matches are always affected by outliers. Starting from this problem, we aim in this project to make a comparison between outlier filtering methods, Adaptive Locally-Affine Matching (AdaLam), and Lowe's ratio test in terms of their influence on the pose-estimation quality and time consumption. AdaLam is a hierarchical method designed to effectively exploit modern parallel hardware for fast and accurate outlier filtering based on local affine motion verification with a sample-adaptive threshold. Lowe's ratio test matches key points based on distance measurements by comparing the distance of the two nearest neighbors for identifying distinctive correspondences. We have also applied two methods to extract key points, SIFT and ORB, and studied their effect on the outlier filters. To perform experiments, two methods were used in the pose-estimation pipeline and the conclusion is based on the quality metrics of the computed transformation matrix between a pair of images. Images pairs for the dataset were constructed from the TUM RGB-D Dataset. We have demonstrated that SIFT is better than ORB in terms of the total number of key points generated. We have also shown that AdaLam is better than Lowe's ratio in terms number of correct matches and speed.**

**Keywords— Feature matching, image matching, key points, outlier filters, computer vision, SIFT, ORB, Low's ratio, AdaLam.**

## I. Introduction

Local feature matching is an essential problem in many computer vision applications in different fields, such as robotics (Simultaneous Localization and Mapping [1], Structure from Motion [2], etc.), object tracking, face matching, etc. Image matching is used in data association problems in robotics, which aims to help the robot to detect whether it exists in a new position or revisits the already existing one. A primary step for image matching is key point detection and descriptors computation. This problem can be approached with various methods [3, 4] which compute and extract features from images and provide high-dimensional descriptors for each of them.

Paper received 25 June 2023.

Mohammed Hammoud, PhD student, National Research University of Electronic Technology (MIET) (e-mail: hammoudmsh93@gmail.com);

Melaku Getahun, Master student, Skolkovo Institute of Science and Technology, (e-mail: mnegussie80@gmail.com);

Sergey Lupin, Professor, National Research University of Electronic Technology (MIET), (e-mail: lupin@miee.ru).

These points with descriptors are later used for matching similar parts in an image pair. However, the resulting set of correspondences is contaminated with a huge amount of outliers–observations that differ strongly from the other data points in the sample of the population – which may be due to various reasons, such as limitations in the descriptors.

To tackle this problem, outlier detection and filtering may be applied to the set of key points to enhance the matching results. For this task, a number of methods have been proposed in recent years [3, 4] that exploit different insights to define rules to provide qualitative filtering. In this project, we aim to study the influence of different types of outlier filters on the results of the pose estimation problem. We have chosen, two outlier filtering methods: mutual Lowe Ratio and AdaLam as examples of a well-proven approach and a novel solution. We embedded this method into a pose estimation pipeline with the use of SIFT and ORB methods for key points extraction and descriptors generation for additional comparison. We used OpenCV methods to estimate the translation vectors and rotation matrix for a pair of images from filtered correspondences. A dataset of image pairs for the experiments was constructed from TUM RGBD dataset [5, 6], which provides a number of image sequences in indoor environments. To calculate an error in our estimation we used two metrics: one for translation direction and another for rotation matrix. Metrics were computed for each pair of images in the dataset and these results, coupled with time consumption scores, are presented in the Experiments section.

## II. Key point detection

Until this moment a huge variety of image-matching methods have been presented. Estimation criteria are different [3], some of them are:

**Speed per frame** – The required time (in ms) for the feature detection of the single frame.

Speed per keypoint – detection time for a single key point (total time divided by the number of detected key points).

**Percent of tracked features** presents percent of successfully tracked features from the original to transformed image. In an ideal situation, the value of this mark should be near 100%.

**Average tracking error** – this is the average distance between the position of tracked feature and their calculated position on the transformed frame. This mark indicates the

accuracy of the feature detection. Larger values indicate a larger number of false positive tracking or “drift” of feature points among frames.

**Features count deviation** – the difference between the number of key points on the reference frame and the number of detected key points on the transformed frame divided by the number of key points on reference frame. Helps estimate how slight exposure changes affect feature detection.

**Average detection error** – the average distance between the nearest key points on the original and transformed frame.

As mentioned before, there are different algorithms used for keypoint detection, such as:

**SIFT** (Scale Invariant Feature Transform)

**SURF** (Speeded Up Robust Feature)

**BRISK** (Binary Robust Invariant Scalable Key points)

**BRIEF** (Binary Robust Independent Elementary Features)

**ORB** (Oriented FAST and Rotated BRIEF)

In [4] they conclude that ORB requires less time than SIFT and SURF, as shown in Table 1.

Table 1. Time comparison: ORB, SURF and SIFT

| Detector | ORB | SURF | SIFT |
|---|---|---|---|
| Time per frame (ms) | 15.3 | 217.3 | 5228.7 |

Fig. 1 shows the difference between methods listed from the time point of view and Fig. 2 compares them using the number of detected key points [3].

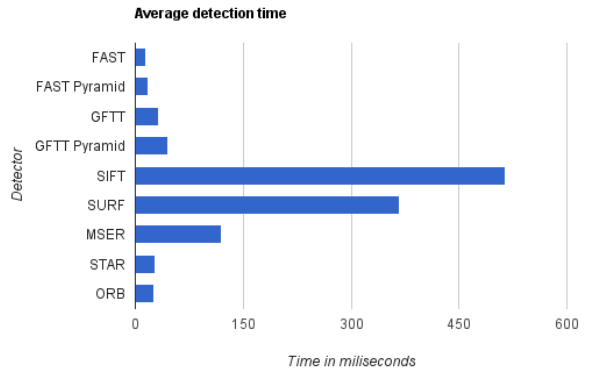



Fig. 1 Average detection time

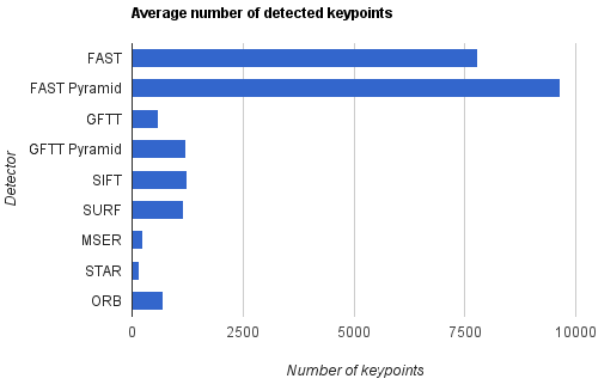



Fig. 2 Average number of detected key points [3].

Some methods for distinct feature detection have high performance if the images are un-scaled and un-rotated like Harris Detector which is good for rotation variance but it does not perform well if the image is scaled. And in the real world images come with various appearances rotated in different directions and scaled. SIFT [7] is both scale invariant and rotation invariant. It performs well and efficiently no matter how the image is rotated and scaled. And the other method for distinct feature detection used is ORB [4], a combination of two methods called FAST (Features from Accelerated Segment Test) and BRIEF (Binary Robust Independent Elementary Features). The key points are determined by considering pixel brightness around the distinct feature area faster than SIFT but its accuracy and number of key points detected are lower when compared to SIFT.

## III. Outlier filters

### A. *Lowe’s Ratio Test*

This method depends on filtering the results of the Nearest Neighbors (NN). The idea behind it is to calculate the ratio between the minimum two distances for each key point. A key point can pass this filter only if this ratio is lower than a threshold we have already specified, in our case it was r = 0.95. The test can be written as in the following equation (1):

$$\frac{\|d_i^1 - d_j^2\|}{\|d_i^1 - d_k^2\|} < r, r \in [0,1] \quad (1)$$

Where $d_i^1$ are the descriptors of the key point $i$ from the first image, $d_j^2$ are the descriptors of the key point j from the second image that achieve minimum distance, and $d_k^2$ is the descriptors from the second image that achieves the second minimum distance, and r is the threshold.

**Mutual NN**: To achieve the best outcome from Lowe’s ratio test we added an additional condition. The condition is to verify whether the key points from each image are referring to each other, other than that we exclude these key points. The condition can be written as in the following equation (2):

$$\widetilde{m}_i^1 \to j \cap \widetilde{m}_j^1 \to i \quad (2)$$

Where:

$$\widetilde{m}_i^1 = \underset{j \in 0,\dots,n_1-1}{arg\min} \|d_i^1 - d_j^2\|$$

$$\widetilde{m}_i^2 = \underset{i \in 0,\dots,n_1-1}{arg\min} \|d_j^2 - d_i^1\|$$

In this paper, we have applied the mutual NN and Lowe’s ratio as our first outlier filtering method.

### B. *Adaptive Locally-Affine Matching (AdaLam)*

In [8] a new time-efficient hierarchical approach for outlier filtering was introduced. Here, the filtering problem is addressed in three steps: selection of confident and well distributed matches, called seed point; selection of compatible correspondence per seed point; local affine consistency verification in the neighborhood of each seed point. The first step exploits a ratio test as a confidence score to rate the candidates for seed points – points with the highest score in the area of radius R, which will serve as a hypothesis for rough region correspondences. Each comparison is performed independently on Graphical Processing unit (GPU). After we extract seed points we have a set of regions with correspondences, which define the search space for affine transform A. For each pair of corresponding regions, we keep only points that satisfy the constraints on local consistence and induce transform

similarity which is ensured to be reliable by independent thresholds. If we define seed point correspondence, as $S_i = (x_1^{S_i}, x_2^{S_i})$ with a similarity transformation $(\alpha^{S_i} = \alpha_2^{S_i} - \alpha_1^{S_i}, \alpha^{S_i} = \sigma_2^{S_i}/\sigma_1^{S_i})$ from its local feature frame, presented as orientation component $\alpha^{S_i}$ and scale component $\alpha^{S_i}$ and candidate correspondence $(p_1, p_2) = \big((x_1, d_1, \sigma_1, \alpha_1), (x_2, d_2, \sigma_2, \alpha_2)\big)$, the formula in equation (3) demonstrates the constraints for correspondences inside each region:

$$\left\|x_1^{S_i} - x_1\right\| \le \lambda R_1 \cap \left\|x_2^{S_i} - x_2\right\| \le \lambda R_2 \qquad (3)$$
$$|\alpha^{S_i} - \alpha^p| \le t_\alpha \cap \left|ln\left(\frac{\alpha^{S_i}}{\alpha^p}\right)\right| \le t_\alpha$$

Here $R_1$ and $R_2$ are the radii used to spread seed points respectively in images $I$ and $I_2$, and λ is a hyper-parameter. The last step, an affine verification, is performed independently for each set of region correspondences and here authors approach the problem of inlier selection by the classical RANSAC [9] framework with some modifications and fixed number of iterations. In the proposed algorithm, at each iteration, a residual is assigned to each correspondence to measure a deviation of the estimated affine transformation. To determine whether the observed key point is an outlier a threshold should be applied to the residual between projected point and correspondence. However, in this case instead of threshing directly on the error score, the authors' threshold on the statistical significance of an inlier set against the null hypothesis of uniformly scattered outliers. To implement this, confidence in the decision on inlier points is defined as a ratio between the number of inliers actually found and the number of inliers that would be found under the outlier-only hypothesis, refer to equation (4).

$$c_k(\mathcal{R}) = \frac{P}{E_{H_0[P]}} \qquad (4)$$

In equation (4), k is an index of observed keypoint, $r_k$ is a corresponding residual $r_k = \left\|A_i^j x_k^1 - x_k^2\right\|$, R is a set of all residuals, $R_2$ is a sampling radius in the second image and P is a number of samples which have lower residual than the mapped keypoint k. The whole implementation is claimed to effectively exploit modern parallel hardware leading to fast image processing even with a high amount of key points on a modern GPU.

## IV. Dataset

The dataset was constructed from the TUM RGB-D SLAM Dataset and Benchmark [5], which contains a number of subsets with rdb and depth image sequences with corresponding annotations. We used a freiburg1 xyz sequence in which the Kinect scanner was pointed at a typical desk in an office environment and moved along it. This dataset contains around 800 images with annotations – translation vectors and rotations, written in quaternions in the ground coordinates–that define transfer from one frame to the next one. The correspondences of the annotation files and images are provided through timestamps in the filenames. Fig 3 shows some images from the used dataset. To compose a pair of an image and annotation we searched for the close timestamps in the image file name and annotation filename. After that, we defined a step to compose image pairs from the sequence. For simplicity, we kept 17 pairs which were constructed for a step of 10.

Fig 3 Example of a dataset image

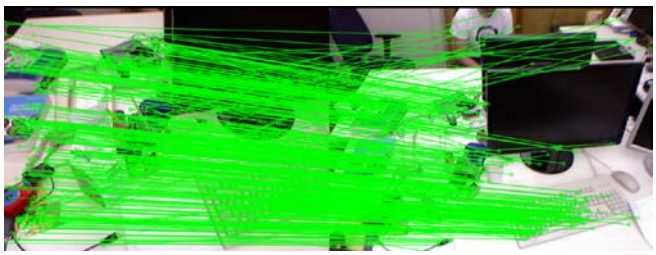
## V. Results

We have applied SIFT and ORB methods to obtain the key points from each image. Then both previously mentioned filters were used to obtain the required filtered data associations, Lowe's ratio with mutual NN, and AdaLam. In the next step, a pose-estimation algorithm was applied for each method to visualize the effectiveness of each filter.

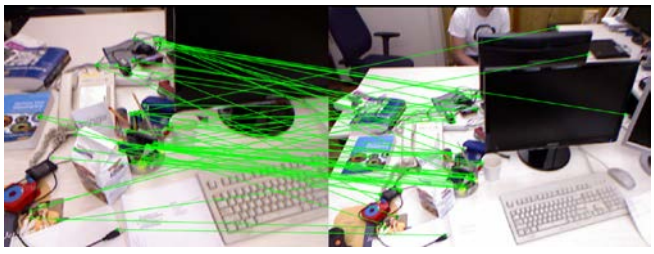
### A. *Lowe's ratio with mutual NN*

The results of applying SIFT and ORB methods to obtain the key points and then applying Lowe's ration filtering mutual NN can be seen in Fig 4. As we can see from Fig 4 that the number of key-point generated by SIFT is much more than the number of key points generated by ORB. Furthermore, we can observe that this filtering method doesn't filter all the wrong matches. But yet, it gives a good number of correct matches compared to incorrect ones.

### B. *Adalam*

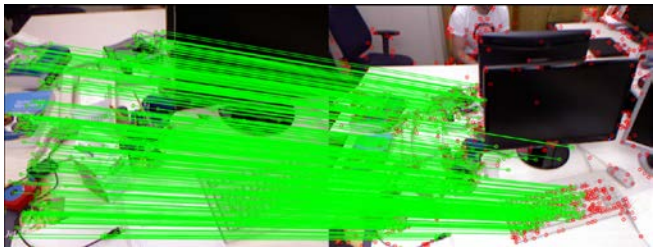


(a) Mutual Lowe's Ratio filter on SIFT

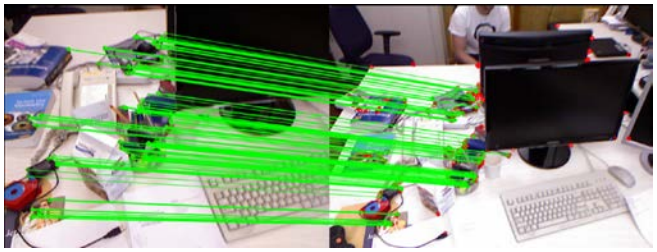


(b) Mutual Lowe's Ratio filter on ORB

Fig. 4 Applying Lowe's Ratio with mutual NN on a pair of images.



(a) AdaLam filer on SIFT



(b) AdaLam filter on ORB

Fig 5 Applying AdaLam on a pair of images.

In all in the pairs generated with 100% rating success. The results also showed that the AdaLam method is 10 times faster than Lowe's ratio. When applying both filters method on 20-pair of images consumed 3.5(sec) and 0.35(sec) for Lowe's ratio and AdaLam respectively.

### C. Pose-estimation

For this part, we have adapted the OpenCV library in Python. Taking advantage of some useful functions in it along with the intrinsic matrix provided in the dataset.

A ground truth annotation was provided in the dataset in the format $t_x, t_x, t_y, t_z, q_x, q_y, q_z, q_w$, where $[t_x, t_y, t_z]$ is the position of the optical center of the color camera with respect to the world origin as defined by the motion capture system. And $[q_x, q_y, q_z, q_w,]$ is the orientation of the optical center of the color camera in the form of a unit quaternion with respect to the world origin as defined by the motion capture system [6]. The orientation was converted to rotation matrices to simplify the procedure. The OpenCV library gives us the rotation matrix between two images and the translation vector between them. Noting that the translation vector is normalized so all we can get is the direction of movement but not the distance of translation.

### D. Error Metrics

A comparison between the rotation and translation vector was conducted to verify our results. To achieve this, two metrics were adapted. The first matrix is to measure the error in rotation. Which means comparing the two rotation matrices. The one resulted from the OpenCV library and the one calculated from the dataset. For each pair of images, we considered that the left image is the starting point and a real rotation resulted in the second image, $im_2$ and an estimated rotation resulted in $im'_2$.And our first metric calculates the rotation between $im_2$ and $im'_2$. The equation to calculate that is given as the following equation (5):

$$R_{error} = {}^{C_1}_{C_2}R \, {}^{C_1}_{C'_2}R \tag{5}$$

Where ${}^{C_1}_{C_2}R$ is a real rotation between the first camera and the second camera, and ${}^{C_1}_{C'_2}R$is the estimated result from OpenCV. The second metric was to calculate the angle between the translation vector calculated from the dataset and the normalized translation vector we got from the OpenCV. To do that, the definition of dot vector was applied to calculate the |θ| between them, as presented in the following equation (6):

$$\theta_{direction\ error} = cos^{-1} \frac{\overrightarrow{\frac{1}{2}t} \ \overrightarrow{\frac{1}{2'}t}}{\left|\overrightarrow{\frac{1}{2}t}\right| \left|\overrightarrow{\frac{1}{2'}t}\right|} \tag{6}$$

Where $\overrightarrow{\frac{1}{2}t}$ is the real translation vector from the first image to the second image written in the ground coordinate system. And $\overrightarrow{\frac{1}{2'}t}$ is the normalized estimated translation vector got from OpenCV that translates from the first image to the second image and is written in the ground coordinate system. The result of applying these two explained metrics on the pairs of images can be seen in Table 2 and Table 3. Table 2 shows the error between the estimation of the translation and its real value, calculated as explained before, depending on the equation *(6*). While Table 3 on the other hand shows the error in rotation between the estimated second image and the real second image depending on the equation (5).

Table 2. The error in the direction of translation

| AdaLam error (degrees) | Lowe's ratio error (degrees) |
|---|---|
| 121.86 | 118.86 |
| 117.38 | 68.81 |
| 66.89 | 108.35 |
| 111.93 | 113.82 |
| 114.28 | 85.34 |
| 63.79 | 120.08 |
| 73.31 | 64.20 |
| 74.86 | 123.56 |
| 125.32 | 71.88 |
| 114.62 | 114.51 |
| 98.59 | 61.84 |
| 131.50 | 115.28 |

Table 3. The error rotation

| AdaLam error (degrees) | Lowe's ratio error (degrees) |
|---|---|
| 5.15 | 5.45 |
| 2.15 | 1.24 |
| 5.91 | 3.98 |
| 2.04 | 2.55 |
| 6.33 | 7.87 |
| 10.51 | 11.02 |
| 5.17 | 5.31 |
| 2.67 | 2.96 |
| 9.48 | 8.68 |
| 3.59 | 3.53 |
| 3.43 | 3.22 |
| 2.08 | 2.08 |
| 2.67 | 2.96 |
| 9.48 | 8.68 |
| 3.59 | 3.53 |
| 3.43 | 3.22 |
| 2.08 | 2.08 |

As we can see from Table 2 and Table 3 that the error in estimation is very big for both filters. This is probably because we are depending on the OpenCV library for pose estimation. This library does not have optimization before calculating the rotation and translation norm between two

images.

## VI. Conclusion and future work

In this paper, we presented a comparison between two outlier filters, Lowe's ratio with mutual NN, and AdaLam, in their effect on pose-estimation quality. We have shown at first the number of key points generated by SIFT and ORB. Then we presented the precision of each outlier filter and the error in matching between key-point. And in the end, we compared these two filters for pose estimation. We have got huge errors concerning the results, especially regarding the direction of the translation vector.

In our future work, we want to apply the OpenGV library instead of OpenCV for pose-estimation, because the first library performs an optimization procedure on the two images while calculating the rotation and the normalized translation vector. Additionally, we would like to include a third outlier filter, such as ORB-SLAM, and make a comparison of the pose-estimation effect with the other two studied filters. And additionally, we also want to test our program on other datasets to verify the correctness of our work.

## REFERENCES

[1] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (slam): part ii," IEEE Robotics Automation Magazine, vol. 13, no. 3, pp. 108–117, 2006.
[2] R. I. Hartley and P. Sturm, "Triangulation," Computer Vision and Image Understanding, vol. 68, no. 2, pp. 146–157, 1997.
[3] Eugene Khvedchenya, https://computer-vision-talks.com/2011-07-13-comparison-of-the-opencv-feature-detection-algorithms
[4] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," Proceedings IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011, pp. 2564–2571.
[5] J. Sturm, N. Engelhard, F. Endres, W. Burgard, D. Cremers, A benchmark for the evaluation of RGB-D SLAM systems, in: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012, pp.573–580. doi:10.1109/IROS.2012.6385773.
[6] Dataset from Computer Vision Group TUM School of Computation, Information and Technology Technical University of Munich, https://vision.in.tum.de/data/datasets/rgbd-dataset/download.
[7] Y. Zhao, Y. Zhai, E. Dubois, and S. Wang, "Image matching algorithm based on sift using color and exposure information," Journal of Systems Engineering and Electronics, vol. 27, pp. 691–699, 2016.
[8] L. Cavalli, V. Larsson, M. R. Oswald, T. Sattler, and M. Pollefeys, "AdaLAM: Revisiting handcrafted outlier detection," ArXiv, vol. abs/2006.04250, 2020.
[9] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," Commun. ACM, vol. 24, pp. 381–395, 1981.