SIFT Feature Point Matching Based on Improved RANSAC Algorithm

Guangjun Shi, Xiangyang Xu, Yaping Dai
School of Automation
Beijing Institute of Technology
Beijing, China
E-mail: shiguangjun.111@163.com, xxy1970@126.com, daiyaping.bit@gmail.com

***Abstract*—When matching the SIFT feature points, there will be lots of mismatches. The RANSAC algorithm can be used to remove the mismatches by finding the transformation matrix of these feature points. But when the data space contains a lot of mismatches, finding the right transformation matrix will be very difficult. What's more, the probability of finding the error model is very large. Aiming at solving the problem, this paper proposed an improved RANSAC algorithm. Before using the RANSAC algorithm, we removed parts of the error feature points by two methods, one is eliminating features not belonging to the target area and the other is removing the crossing points. The two methods aimed to improve the proportion of feature points matched correctly. Experiments showed that, the improved RANSAC algorithm could find the model more accurately, improve efficiency, and make the feature point matching more accurately.**

Keywords- SIFT, key point matching, improved RANSAC

I. INTRODUCTION

SIFT [1, 2, 3] (Scale Invariant Feature Transform) is a feature point detection and matching algorithm proposed by Lowe in 1999, and perfectly summarized in 2004. The SIFT features are not only invariant to image scaling, translation, and rotation, but also partially invariant to illumination changes and affine. The SIFT algorithm based on feature recognition and matching is used in many fields, such as object recognition [4, 5], image mosaic, mobile robot localization and map building , and so on [6, 7].

Lowe used the method of comparing the distance of the closest neighbor to that of the second-closet neighbor to match the key point. When the ratio is less than the threshold, it can be considered to be the correct matching. When reducing the threshold, matching points will decrease, but little mismatches. When raising the threshold, matching points will increase, but there will be a lot of mismatches. Usually we want feature points matched correctly the more the better. In this case we need to increase the matching threshold, but following there will be a lot of mismatches. In order to solve this problem, actually we need to find a robust estimation method, which should estimate the parameters to build the model to classify the data containing many outlier data. The researchers recommended some strong robust estimation method [8, 9]; one of the most representatives is the RANSAC [10] algorithm. But when the data space contains a lot of mismatches, finding the right transformation matrix will be very difficult, and the computer need lots of time to calculate the parameters. The efficiency will be substantially low; what's more the probability of finding the error model is very large. Aiming at solving the problem, this paper proposed an improved RANSAC algorithm. Before using the RANSAC algorithm, we can remove parts of the error feature points by two methods, one is eliminating features not belonging to the target area and the other is removing the cross points. The two methods aimed to improve the proportion of correct matching feature points. Experiments showed that, the improved RANSAC algorithm could find the model more accurately, improve efficiency, and make the feature point matching more accurately.

II. IMPROVED RANSAC ALGORITHM

A. Introduction of RANSAC Algorithm

RANSAC (Random Sample Consensus) is a method of robust estimation. It was proposed by M. A. Fischler in 1981. It calculates the model parameters mainly based on random voting principle. RANSAC algorithm is of strong robustness, it can tolerate the condition that the data space containing more than half of the outliers and can effectively deal with multiple structure data.

There are the following assumptions:

(1) ω = the number of inlier / the number of outlier, namely ω is the probability of a feature point selected as the correct feature point.

(2) If we need to select n points to estimate the model, the probability of the n points in inlier is ω^n. Then the probability of at least one point in outlier is $1-\omega^n$, which shows that we estimate a bad model.

(3) If we estimate the model for k times, the probability of the n points never in inlier is $(1-\omega^n)^k$.

(4) P stands for the probability of estimating a wrong model, then

$$P=(1-\omega^n)^k \tag{1}$$

$$k=\frac{\ln P}{\ln(1-\omega^n)^k} \tag{2}$$

978-0-7695-5011-4/13 $26.00 © 2013 IEEE
DOI 10.1109/IHMSC.2013.119

From the formula (1) and (2) we can find, when P is constant, k decreases with the increase of ω. In order to make the probability of finding error model as small as possible, it is necessary to increase the number of times of sampling. So when the data space contains huge amount of outlier data, the RANSAC algorithm is inefficient, and may even find the wrong model, as in Fig. 7.

B. Two Methods of Improving the RANSAC Algorithm

From the above analysis, in order to improve the accuracy and efficiency of the RANSAC algorithm, we can increase the parameter ω, that is, increasing the proportion of the inliers, and decreasing the proportion of outliers. This passage is based on this idea to improve the SANSAC algorithm.

The passage used two methods to improve the SANSAC algorithm, shown below:

- Method One: Find the target area and remove the feature points not in the target area.

Usually the picture we want to match contains the target object and background. As shown in Fig. 1-b, the train is the target objet and others are the background. We name the area where the train locates in as the target area. Except the target object can generate feature points, the background can also generate many feature points and some of them will match successfully, which obviously are mismatches, as shown in Fig. 3. So these feature points not belonging to the target area can be removed in advance. Fig. 3 shows that there are only a few of feature points around the area of these points matching successfully but not belonging to the target area. So we can calculate the number of every point's around area containing feature points. When the number less than the threshold we set, we think the point doesn't belong to the target area and remove it.

It is supposed that feature point $A(x, y)$ is one of the matching feature points. Calculate the number of matching feature points in the rectangular region whose center is (x, y), width is "a" and height is "b" ("a" and "b" are set by us). If the number less than the threshold value, $A(x, y)$ is supposed the isolated point not in the target area. Then remove $A(x, y)$. Examine each of the matching points.

- Method Two: Remove the crossing feature points.

As shown in Fig. 6, there should be no crossover between the connections of the feature points matching each other. But there are a large number of cross-lines as shown in Fig. 3. By comparing Fig. 3 with Fig. 6, it can be found that if a line crosses with large part of the lines, the point should be a mismatch. Therefore, we can set a threshold value, then calculate the number of the line crossing with other lines, when the number is bigger than the threshold value, we can remove the point.

III. Combination of SIFT Feature Points Matching and the Improved RANSAC Algorithm

A. Extract Image's SIFT Feature Points

SIFT feature points extraction mainly includes four parts, shown below:

(1) Scale-space extreme detection

SIFT algorithm selects scale space extreme points as candidate feature points. The image $I(x, y)$ scale space is defined as:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{3}$$

Detect the extreme in the result of DoG (Difference of Gaussian) and image convolution:

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{4}$$

Where $*$ presents two-dimensional convolution; $G(x, y, \sigma)$ presents Gaussian function; σ presents standard deviation of normal Gaussian distribution.

(2) Key point localization

Construct the Taylor expansion in scale space of DoG function:

$$D(X) = D + \frac{\partial D^T}{\partial X} X + \frac{1}{2} X^T \frac{\partial^2 D}{\partial X^2} X \tag{5}$$

Then solve the formula:

$$\hat{X} = -\frac{\partial D^T}{\partial X} \left(\frac{\partial^2 D}{\partial X^2} X\right)^{-1} \tag{6}$$

$$D(\hat{X}) = D + \frac{1}{2} \frac{\partial D^T}{\partial X} X \tag{7}$$

(3) Orientation assignment

Assign a main direction for each feature point, containing gradient magnitude $m(x, y)$ and gradient direction $\theta(x, y)$:

$$m(x, y) = [(L(x, y+1) - L(x, y-1))^2 + (L(x+1, y) - L(x-1, y))^2]^{1/2} \tag{8}$$

$$\theta(x, y) = \arctan \frac{L(x, y+1) - L(x, y-1)}{L(x+1, y) - L(x-1, y)} \tag{9}$$

(4) Key point descriptor

Divide the image region around the critical point to blocks, calculate the gradient histogram in each block, and generate a unique 128-dimensional vector.

B. Use the Improved RANSAC Algorithm to Remove Mismatches

- Step One: increase the proportion of correct points.

We use the two methods mentioned above to remove parts of the error feature points, and increase the proportion of correct matching feature points.

- Step Two: determine a transformation function.

Determine a transformation function according to the projection transformation relations of two images. This passage uses a projective transformation，shown below:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad (10)$$

- Step Three: find the model.

(1) Randomly select four groups of points from matching points, calculate the parameter of the model as formula (10) shown;

(2) Judge whether the other points belonging to the model (the error is less than a given threshold value), record the number of data belonging to the model;

(3) Judge the number of points in the third part , if it is the biggest, then retain the model, if not abandon the model;

(4) Repeat the second, third and fourth step until the error probability P is smaller than 0.01, then the model is found;

(5) After model found, estimate the model again.

IV. Experimental Results

The experiment used the laboratory highly simulation sand model, and a small train as the target. Network ball machine is used to capture pictures. Software test environment is VC++6.0 and OpenCV.

A. Extract Image's SIFT Feature Points

Extract SIFT feature points of the two pictures shown in Fig. 1. The numbers of SIFT feature points in Fig. 1-a and Fig. 1-b are respectively 69 and 542. Fig. 2-a and Fig. 2-b respectively show the feature points extracted by SIFT algorithm.

a b

Figure 1. Pictures of model and target.

a b

Figure 2. Pictures of the SIFT descriptor.

B. Match SIFT Feature Points

We use the method of comparing the distance of the closest neighbor to that of the second-closet neighbor to match the key point. When the ratio is less than the threshold, it can be considered to be the correct matching. The matching effect is shown in Fig. 3.

In the picture Fig. 3-a, we set the threshold value 0.7, the number of matching points is 24; while in the picture Fig. 3-b, we set the threshold value 0.8, the number of matching points is 39. The two pictures all have mismatches. Comparing the two pictures we can find that when reducing the threshold, matching points will decrease, but little mismatches; when raising the threshold, matching points will increase, but there will be a lot of mismatches.

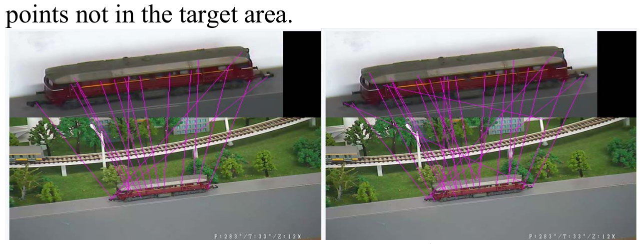
a b

Figure 3. Pictures of matching under different threshold.

C. Match Again Using the Improved RANSAC

- Experiment One: remove points not in the target area.

After using the first method, there are 16 matching points in Fig. 4-a, 8 less than in Fig. 3-a; there are 24 matching points in Fig. 4-b, 15 less than in Fig. 3-b. Through comparing Fig. 4 with Fig. 3, we can find almost all points in the target area. In conclusion the method one can remove the points not in the target area.



a b

Figure 4. Pictures of removing the points not in the target area.

- Experiment Two: remove the cross points

After using the second method, there are 13 matching points in Fig. 5-a, 3 less than in Fig. 4-a; there are 20 matching points in Fig. 5-b, 4 less than in Fig. 4-b. Through comparing Fig. 5 with Fig. 4, some cross points have been removed. In conclusion the method two can remove the cross points.

a b

Figure 5. Pictures of removing the cross points.

- Experiment Three: calculate the parameter of the model

Based on experiment one and two, we have removed large part of mismatches. But there are also some mismatches we can't remove by using the two methods. So it's necessary to find the model to match again. The model is shown in formula (11).

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 0.1717 & -0.8119 & 205.8064 \\ -0.0599 & -0.1176 & 130.4381 \\ -0.0003 & -0.0025 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (11)$$

The matching effect is shown in Fig. 6. From Fig. 6 we can see all the points matching successfully, so we have found the correct model.

a b

Figure 6. Pictures of using improved RANSAC.

Summarize all the experiments and compare the number of feature points remained in every experiment. The result is shown in Table 1. From Table 1 we can find that based on experiment one and experiment two, we have removed large parts of mismatches, and increased the proportion of correct matching feature points. The proportion of correct points increases respectively from 33% to 62% and from 31% to 60%. In this case we can find the correct model more easily and correctly than not using the two methods. As there are large parts of mismatches, it is more likely to find the wrong model. Fig. 7 shows the wrong model using the before RANSAC. In a word the two methods we use actually played their roles.

TABLE I. Number of Feature Points

Threshold	0.7	0.8
Number of Matching Points	24	39
Number After Using Method One	16	24
Number After Using Method Two	13	20
Number After Finding Model	8	12

a b

Figure 7. Pictures of finding wrong model using the before RANSAC.

V. Conclusion

Aiming at solving the problem that when the data space contains a lot of mismatches, using RANSAC algorithm to find the right transformation matrix will be very difficult and the probability of finding the error model is very large, this paper proposed an improved RANSAC algorithm. Before using the RANSAC algorithm, we can remove parts of the error feature points. The paper proposed two methods to remove the error feature points. Experiments showed that, the two methods actually could remove large parts of mismatches, increase the proportion of correct matching feature points. The improved RANSAC algorithm can find the model more accurately, improve efficiency, and make the feature points matching more accurately.

References

[1] Lowe D G. Object Recognition from Local Scale-Invariant Features, Proceedings of the Seventh IEEE International Conference on Computer Vision. 1999: 1150-1157.

[2] Lowe D G. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 2004, 60(2): 91-110.

[3] Lowe, D.G. Local feature view clustering for 3D object recognition, IEEE Conference on Computer Vision and Pattern Recognition. 2001:682-688.

[4] Xingfang Yang, Yumei Huang, Xuzhao Han, Xingang Yang. A Method for Improving Matching Efficiency of SIFT Features. Chinese Mechanical Engineering, 2012, 23(11): 1297-1301.

[5] Fu Weiping, Qin Chuan, Liu Jia, Yang Shiqiang, Wang W en. Matching and location of image object based on SIFT algorithm. Chinese Journal of Scientific Instrument, 2011, 32(1): 163-169.

[6] Gao Jian, Huang Xinhan, Peng Gang. Simplified SIFT feature point detecting method. Application Research of Computers, 2008, 7(25): 2213-2222.

[7] Jian Cao. Image objects representation and recognition. Beijing: Machinery Industry Press, 2012.

[8] Luong Q T. and Faugeras O D. The fundamental matrix: Theory, algorithms, and stability analysis. International Journal of Computer Vision, 1996, 17(1): 43–76.

[9] Beis J and Lowe D G. Shape indexing using approximate, Conference on Computer Vision and Pattern Recognition. 1997: 1000-1006.

[10] Mikolajczyk, K. and Schmid, C. An affine invariant interest point detector, European Conference on Computer Vision. 2002:128-142.