Object Detection Based on SIFT and Clustering Algorithm

Yuan ZHOU¹, Zhi-yan CUI¹,*, Tao WANG¹ and Xin-yu YANG¹

¹Xi’an Jiaotong University, P.R. China
cuizhiyan@stu.xjtu.edu.cn

Keywords: SIFT, Clustering, Object detection.

Abstract. Detecting objects from complex scenes is a classic problem in computer vision. It is the basis of object recognition and image understanding. In this paper, we propose an object detection algorithm based on SIFT feature and clustering algorithm. We first extract feature points from the template and the target image. Then, after three times of clustering and filtering, the rotation, scaling and translation of the object from the template image to the target image are estimated. Finally, using the transformation matrix, we can obtain the position of the object in the target image. Our experiment result proved that the proposed algorithm outperforms RANSAC in effectiveness and stability. Our method also avoids manually setting of thresholds that is required in the RANSAC method.

Introduction

Object detection solves the problem of locating objects in an image. The outcome of an object detection system is usually a bounding box that indicates the location and the approximate outline of the object [1]. Object detection is the basis of object tracking, image segmentation and understanding. It is widely applied in the fields such as video surveillance, face recognition [2], and information retrieval [3,4].

Traditional object detection method uses SIFT to extract feature points and the RANSAC algorithm to match the points and estimate the transformation matrix [5]. But this method has disadvantages such as running time uncertainty and requiring manual setting of some parameters. To overcome these disadvantages, we use clustering algorithm instead of RANSAC. Our method guarantees a stable running time, provide higher accuracy results, and avoid manual setting of thresholds.

Next, we first introduce the related work of object detection, then explain our method, and finally shows the experiment results and the evaluation of our method.

Related Work

Object detection algorithms can be divided into two categories: detection algorithm based on global and local feature of the object. The global algorithms detect the object by global descriptors such as histogram of oriented gradient [1], edge orientation histogram [6] and deformable parts models (DPM) [7]. Histogram of oriented gradient [1] is to calculate the gradient of all pixels in a window, based on which the gradient descriptor is computed. This method is mainly used in the field of pedestrian detection and face recognition. But as this descriptor is not rotation invariant, it cannot be directly applied to identify the object in different angles. It also has difficulty to deal with partially occluded situation because the global descriptor can be seriously affect by the occlusion. Another type of algorithm is based on local feature of the object. The most widely used algorithm is SIFT (Scale-Invariant Feature Transform) [8]. The SIFT feature is scaling and rotation invariant. SIFT behaves well in the case of partial occlusion. There are many different versions of SIFT. Two popular ones are PCA-SIFT [9] and SURF [10]. The PCA-SIFT descriptor vector is reduced from 128 to 20 dimensions to improve the computation efficiency. SURF use the hessian matrix instead of the gauss difference pyramid to further improve the speed of the operation. According to Luo et al. [11], SIFT
gets the best results for scaling and rotation changes comparing with PCA-SIFT and SURF. We choose SIFT for feature extraction in our method.

Many methods are proposed for matching two sets of feature points. Philbin et al. [12] uses geometric test method. Zhou et al. [13] uses the spatial position relation between the feature points. M-estimators proposed by Jiun-Hung et al. [14] and RANSAC (Random Sample Consensus) proposed by Fischler et al. [15] can also be used to do the matching. RANSAC is a method that first use a set of data to calculate the initial mathematical model, and then iteratively update the model until converge. By using the RANSAC method, we can get the transformation matrix from one image to another. Because the initial inlier points are randomly selected, the RANSAC algorithm only guarantee a reasonable result when the iteration time is large enough. Although the RANSAC algorithm is very popular, its computation time is uncertain because the computation time grows exponentially when the initial parameters increases.

RANSAC and SIFT algorithms are often used together for object detection. First, the feature points are extracted. Then the feature points are matched between the two images by using RANSAC. In this paper, we use clustering method instead of RANSAC to achieve the final match. Our method outperforms the RANSAC in results accuracy. Additionally, the running time of our algorithm is stable.

Our Method

In this section we introduce our method. Firstly, we extract the feature points from both the template image and the target image, and initially match the feature points between the two images. Secondly, the matching of feature points was filtered by three times of clustering. Thirdly, use the final matched feature points, we compute the transformation matrix to get the final result.

Calculation and Initial Matching of Feature Points

First, the classic SIFT algorithm is used to extract feature points. To match the feature points of the template image and that of the target image, for each feature point in the template image, we compare it to all points in the target image. More specifically, Euclidean distance is used to measure the distance between the SIFT features of the point pairs. We validate the match by computing the ratio between the nearest distance and the second nearest distance [8]. If the ratio is larger than threshold 0.8, the match is unlikely to be a correct match so we ignore the match. According to [8], this method can eliminate 90% of the error matching when 5% of the correct match is abandoned.

Filtering of Matching Points Based on Clustering

We assume that the objects are under the transformations including translation, rotation and scaling. By estimating the parameters of these three transformations, we can find the location of the object in the target image. In the above chapter, we have computed feature points in both images and found the corresponding point pairs. Theoretically, if the matching between feature points are perfect, we can compute the transformation matrix by any two matching pairs.

But we cannot guarantee the matching pairs are 100% correct. This is because the locations of feature points on the object can change because of different environment lighting, background color, or even occlusions on the object. To improve the precision of transformation estimation, we need to remove the obviously wrong matching pairs. We apply a census on the estimation results -- estimate the transformations by all matching pairs, and keep the results that appear most frequently. More specifically, to remove the obviously wrong matching, we firstly compute the transformations of all matching points, and find the dominant transformation by clustering. The largest cluster correspond to the dominant transformation is selected. Then the transformation results that are not in the dominant cluster are considered as wrong result, so the corresponding matching pairs are removed. Next we explain the details of our method.
For convenience, we denote the template and target image as A and B respectively.

(1) Rotation: Let’s assume that there are \( n_t \) pair of initial matching points are computed by SIFT which is represented by \( (a_i^0, b_i^0), i = 1, 2, 3 \cdots n_t \). We define the direction of each points as the SIFT dominant direction. Then the rotation angle between the directions of feature points \( b_i \) and \( a_i \) can be computed by:

\[
\text{angle}_i = \text{angle}(b_i^0) - \text{angle}(a_i^0).
\]

By calculating the rotation angle between all pairs of matching points, a sequence of \( \text{angle}_i \) is obtained. Then we use K-means cluster algorithm to divide the angles into three categories. The category containing the most points is the dominant category and it represents the rotation of the object. We keep the matching points in the dominant category and delete others. Assuming \( n_z \) (\( n_z < n_t \)) matching points are left, we can get remaining matching points \( (a_i^1, b_i^1) \) \( i = 1, 2, 3 \cdots n_z \).

(2) Scaling: Though there is scale in a SIFT points, we don’t use this method because it is not precise to use this scale to cluster. Assuming two points \((a_i, a_j)\) in A correspond to two points \((b_i, b_j)\) in B \( (a_i \text{ matches to } b_i \text{ and } a_j \text{ matches to } b_j) \). Then the scaling of the object from A to B can be estimated by the ratio between the length of the \( (a_i, a_j) \) and \( (b_i, b_j) \):

\[
scale_{m_i} = \text{dis}(b_i^1, b_j^1) / \text{dis}(a_i^1, a_j^1).
\]

\( m = 1, 2, 3 \cdots (n_z (n_z -1))/2 \). Distances between all pairs of points in A is calculated, and so as B. Then we can compute a sequence of scaling parameter \( scale_{m_i} \). Again the K-means clustering is used to divided it into three categories, and the dominant category containing the most points can be got. The matching points corresponding to the dominant category will be retained, and the others are removed. After the second filter of the matching points, we get sequences \( (a_i^2, b_i^2) \), \( i = 1, 2, 3 \cdots n_z \) (\( n_z < n_z \)).

To reduce the computation complexity, we can randomly sample point pairs within each image instead of considering all pairs. According to our experiment results, when the sampling number is set to \( 3(n_z -1) \), the results become stable. So we set sample number to \( 3(n_z -1) \) in our experiments.

(3) Translation: Denoting the mean of the dominant rotation parameter category as \( \alpha \), and the mean of the dominant scaling parameter category as \( \text{scale} \), we can compute the coordinates of rotated and scaled feature points by:

\[
(x, y) = (x_i, y_i) \times \begin{bmatrix} \text{scale} & 0 \\ 0 & \text{scale} \end{bmatrix} \begin{bmatrix} \cos \alpha & \sin \alpha \\ -\sin \alpha & \cos \alpha \end{bmatrix}.
\]

Then the translation vector of the matching pair can be computed by:

\[
\text{trans}_i = (v_i, w_i) - (x_i, y_i)
\]

Where \((v_i, w_i)\) is the coordinate of the matching point in target image B.

We apply K-means clustering again to the translation list and find the dominant category. After removing the points which are not in this category, we can get the final matching sequence \( (a_i^3, b_i^3) \) \( i = 1, 2, 3 \cdots n_z \) (\( n_z < n_z \)).

After the above three steps of filtering, we remove the matching points which are likely to be wrong. Then we use the sequence \( (a_i^3, b_i^3) \) to estimate the final transformation matrix. We calculate the rotation, scaling, and translation in turn. The rotation and scaling are not dependent on each other,
so the order can be reversed. The translation is determined by scaling and rotation, so it is necessary to solve the translation in the final step. For each step, we first calculate the corresponding transformation, and use the mean as the final estimated transformation.

**Experiments and Evaluation**

Here we use our method to detect various objects under different illumination or occlusion conditions. For convenience, we show the template image on the left with the green square showing the object position made by manual annotation, and the target image on the right, in which the white box shows the object position, and the red square shows the estimated position. The red lines between two images show the matching between feature points. All our experiments are done with the following computer configuration: Intel(R) Core(TM) i3-3210 CPU 3.2GHz, 4GB, Visual Studio2010.

![Experimental results on indoors objects.](image)

**Experiments on indoor images**

Here we show the results of our method on indoor images containing small objects, such as toys, notebooks, etc.

We can see from the Fig. 1 that no matter how was the objects transformed our method can detect the object in the target image correctly. It also works well when occlusion happens, even when more than half of the object area is occluded by other objects.

**Experiment on Oxford Database**

We also test our method on the affine transformation database from University of Oxford [16]. Experimental results are shown in Fig. 2.

It can be seen that the method works well for outdoor images under different illumination conditions.

223
Comparison and Evaluation

We use 8 pairs of images for the experiments in this section, including the car image in Fig. 2, and the
dinosaur, toy car, notebook, cup and bear example shown in Fig. 1 and additionally an instant coffee
bag example and the transformed classic Luna image.

\[
\text{F1-Measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}
\]

The range of F1 value is [0, 1], where 1 means a better result.

To compare the precision of our method with RANSAC algorithm, we repeat each experiment 5
times and compute the mean of the F1 values. The mean F1 value of RANSAC method is 0.890, and
the value for our method is 0.943. So our method can get a better result compared with RANSAC.

To compare the running times between our method and RANSAC, the experiments above were
repeated 20 times. The standard deviation of running time in each experiment is shown in Table 1.

<table>
<thead>
<tr>
<th>Group</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>Mean value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our</td>
<td>0.192</td>
<td>0.762</td>
<td>0.282</td>
<td>0.761</td>
<td>0.494</td>
<td>0.25</td>
<td>1.279</td>
<td>0.668</td>
<td>0.586</td>
</tr>
<tr>
<td>RANSAC</td>
<td>2.86</td>
<td>1.01</td>
<td>5.109</td>
<td>2.806</td>
<td>2.972</td>
<td>1.088</td>
<td>2.688</td>
<td>1.024</td>
<td>2.445</td>
</tr>
</tbody>
</table>

From Table 1 we can see that our method is better than RANSAC in computing time stability. That
is because that the RANSAC is based on a random selection of samples to estimate the model, which
introduces instability into the running time.

Limitations

Our method only considers the detection of a single object. It cannot identify multiple objects at the
same time. On the other hand, our method doesn’t consider shear or flip of the object, so it cannot
detect objects under shear and flip transformation precisely.
Conclusions

We propose a method to detect object under transformation of rotation, scaling and translation. The key idea of our method is using clustering algorithm to eliminate the error matching between feature points from the template image and the target image. From the experimental results, we can see that our method can deal with both indoor and outdoor images under different illumination and occlusion conditions. We also show that the overall result quality and the stability of running time of our method outperform the RANSAC algorithm. In the future, one possible direction is to extend our method for detecting multiple instances of object. We also interested in detecting the object under more complex transformations such as shear or flipping.

References


