Image Feature Detection as Robust Model Fitting

Dengfeng Chai[1,2] and Qunsheng Peng[1]

[1] State Key Lab of CAD&CG, Zhejiang University, Hangzhoz 310027, China
{chaidf, peng}@cad.zju.edu.cn
[2] Institute of Space and Information Techniques, Zhejiang University, Hangzhou 310027, China

Abstract. In this paper, we describe image feature as parameterized model and formulate feature detection as robust model fitting problem. It can detect global feature easily without parameter transformation, which is needed by Hough Transform methods. We adopt RANSAC paradigm to solve the problem. It is immune to outliers and can deal with image contains multiple features and noisy pixels. In the voting stage of RANSAC, in contrast with previous methods which need distance computation and comparison, we apply Bresenham algorithm to generate pixels in the inlier region of the feature and use the foreground pixels in this region to vote the potential feature. It greatly improves the efficiency and can detect spatially-linked features easily. Experimental results with both synthetic and real images are reported.

1 Introduction

Image feature detection is an important topic in computer vision. Given a gray or color image, edge detection can be applied to detect edges and output an edge image which is a binary image of edge (foreground) pixels and non-edge (background) pixels. Detecting features in this binary image is a difficult problem and is the focus of this paper. The methods proposed up to date are categorized into segment grouping based methods [1, 2] and Hough Transform methods (HT) [3, 4, 5, 6, 7].

Segment grouping based methods consist of two stages: linking foreground pixels into segment elements and grouping these elements into global features. Since the grouping criteria are locally optimal, the performance of detecting global features is poor.

In contrast with segment grouping based methods, HT methods map foreground pixels into parameter space and detect features in parameter space. They consist of voting and searching stages, i.e. mapping foreground pixels into accumulators in parameter space and detecting maximal value in accumulators. Because pixels belong to one feature are mapped to one accumulator, they can detect global features successfully at the cost of great storage for accumulators in parameter space and computation time for voting and searching process. Besides, the spatial relationship of foreground pixels is lost in the voting stage.

P.J. Narayanan et al. (Eds.): ACCV 2006, LNCS 3852, pp. 673–682, 2006.

To save computation time, Probabilistic HT (PHT) [5] selects a pre-selected proportion of the foreground pixels in original image for voting. The time saved depends on the ratio of selected pixels with respect to all foreground pixels. Too small proportion frequently leads to incorrect detection results. To select a proper proportion, a priori knowledge about the image is needed. Progressive PHT (PPHT) [7] requests no a priori knowledge, it selects pixels randomly for voting, removes the foreground pixels from image and un-votes accumulator once a highest peak and corresponding line segment is detected. To alleviate the extra storage requirement, Random HT (RHT) [6] adopts many to one mapping and list structure techniques. The computation time is also saved by these techniques.

Chen and Chung have modified RANSAC and developed Random Line Detection (RLD) [8] and Random Circle Detection (RCD) [9] algorithms. They select three or four foreground pixels respectively to define a line or circle and use the left pixels to vote the defined feature. They can detect features with no need of parameter transformation. But the algorithms are inefficient because of explicit distance computation involved in the voting stage. Besides, the spatial relationship between foreground pixels is not well utilized. Zhang have investigated different parameter estimation techniques and presented a tutorial focusing on conic fitting [10].

Motivated by RLD and RCD, we formulate image feature detection as robust model fitting problem in this paper: treat foreground pixels as data points, use parameterized model to describe the image features (such as lines and circles), and treat feature detection as model fitting. Since the global information is implicated in the parameterized model, it can detect global features easily. We adopt RANSAC [11] to solve to the fitting problem, RANSAC is a robust method and is immune to outliers in the original data points, therefore it can detect feature from image contains multiple features and noise pixels. In the voting stage of RANSAC, instead of checking all foreground pixels to vote the feature, we adopt Bresenham algorithm [12] to generate pixels within *inlier region* of the feature and use foreground pixels in this region to vote the feature. This avoids explicit distance computation and improves efficiency greatly. Besides, it detects features directly in image space without involving parameter transformation, therefore needs no extra time and storage requirement. The successive pixels generated by the Bresenham algorithm are spatially neighboring, this property is easily utilized to detect spatially-linked features.

We formulate image feature detection as robust model fitting problem in section 2, propose the solution in section 3, and then present the detection algorithm in section 4. After that, we show experiment results in section 5 and draw conclusion in section 6.

2 Problem Formulation

2.1 Feature Representation

As shown in Fig 1, there are many foreground pixels in the image, some pixels form line l_1, l_2 and l_3 , some form ellipse e, some form circle c and some form

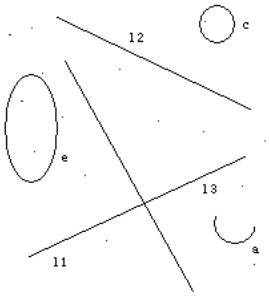

Fig. 1. Image feature detection as robust model fitting

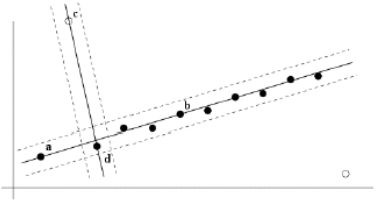

Fig. 2. RANSAC for line fitting: the dotted lines indicate the threshold distance

circle arc a, and others are just noise pixels. The lines, circle, ellipse etc. are image features to be detected. All pixels on one feature satisfy Eq.(1):

$$ax^2 + bxy + cy^2 + dx + ey + f = 0 \tag{1}$$

where a, b, c, d, e, f are free coefficients, so, the features can be described as Eq.(1) with a, b, c, d, e, f being specified.

Eq.(1) is a conic equation describes general curves including circle, ellipse and etc. And these curves are just specific conic with their coefficients meet some constraints. For example, if $a = c, b = 0$, then the conic degrades to be a circle, if $a = b = c = 0$, then the conic degrades to be a straight line. In this paper, we represent image features as equations like that of Eq.(1) and call it as *model representation*. We deal only with image features that can be described by the parameter equation. The model has free coefficients a, b, c, d, e, f and their specified values defines an image feature.

2.2 Image Feature Detection as Robust Model Fitting

Let us assume at first that there is only one line l_1 in the image shown in Fig.(1) and we want to detect it. As shown in subsection 2.1, l_1 can be represented by Eq.(1) with $a = b = c = 0$ and d, e, f being specified. What left to do is to specify the free coefficients d, e, f, it is a well-known *model fitting* problem: fit a model to the pixels so that the distance of the pixels deviated from the model is minimized.

But there are l_2,l_3, etc. together with many noise pixels in the image, fitting a model to all the foreground pixels is meaningless and can not detect the features at all. It is necessary to distinguish pixels which belong to l_1 from other pixels first. Once this is done, the model fitting methods can be applied to fit a model to the distinguished pixels. From the point of view of model fitting, all pixels on line l_1 are inliers to l_1 while other pixels are outliers. The model fitting method must be robust enough to deal with cases there are outliers in the original pixels. It is the nature of *robust model fitting* problem [13].

Since there are many features in the image, it is necessary to carry out the model fitting method repeatedly until all the features are successfully detected.

3 Solution to the Feature Detection

In the previous section, we formulate the image feature detection as robust model fitting problem. There are lots of methods designed to solve this problem [10, 11]. RANSAC can cope with a large proportion of (more than 50%) outliers. As shown in 2.2, since pixels in l_2, l_3 etc. are outliers with respect to l_1, there are usually more than 50% outliers in the original data to be fitted. We adopt the RANSAC algorithm in this paper to solve the model fitting problem.

3.1 RANdom SAmple Consensus

RANSAC does trial repeatedly to find the model. Each trial consists of sampling and voting stages. In the sampling stage, it randomly selects a minimal subset of the original data points and instantiates a model from the subset. In the voting stage, it determines the consensus set of the determined model by distinguishing the set of data points within a distance threshold of the determined model from other points. The termination condition is either a model is found successfully or the number of trials reaches a preset threshold. The algorithm is presented as follow:

1. Set $C_{sample} = 0$, while $T_t > C_{sample}$ do 2-5:
2. **Sampling stage:** Randomly select a sample of s points from original data points and instantiate a model from the selected points,
3. **Voting stage:** Determine the consensus set (set of inlier points) which contains points within a distance threshold T_d of the model,
4. If the size of the consensus set is greater than a preset threshold T_c, report the model and terminate,
5. Let $C_{sample} = C_{sample} + 1$,
6. The largest consensus set is selected as inliers and corresponding model is selected as the final model.

where C_{sample} is the counter for trial number, s is the minimal number of points needed to determine a model. T_d depends on the required fitting precision, T_c is a function of number of inliers and T_t is specified by:

$$T_t = \lg(1-p)/\lg(1-\epsilon^s) \tag{2}$$

where p is the probability that at least one random sample is free of outliers, it is always chosen as 0.99, ϵ is the proportion of inliers.

Fig. 2 illustrates how RANSAC fit a line to the data points. It randomly selects 2 points to define a line, points between the two dashed lines parallel with the defined line are within a distance threshold to the line and form the consensus set. As shown, the size of consensus set of line (a, b) is 10 while that of line (c, d) is 2, so, RANSAC selects line (a, b) as the fitting result at last.

3.2 Sample Minimal Set

It might be thought that it would be preferred to use more than minimal subset to instantiate a model as RLD [8] and RCD [9] do, because a better estimate of the model would be obtained from them, and the measured support would reflect the true support more accurately. However, this possible advantage in measuring support is generally outweighed by the severe increase in computational cost incurred by the increase in the number of trial.

Because there are often lots of pixels in the image, it is computationally infeasible to try every possible sample in the sampling stage. In fact, it is unnecessary to enumerate all the possible samples exhaustively. Instead the necessary number of samples T_t is chosen sufficiently high to ensure that at least one of the random sample of s points is free from outliers with a probability of p. Eq.(2) shows the relationship between T_t and p, ϵ, s. Given an image, the ϵ is constant with respect to the feature to be detected, the p is also constant in the detection process (it is always chosen as 0.99), so, T_t increases exponentially with s. Tab. 1 shows an example of T_t for given s and e. As shown, the necessary number of trials increases dramatically with s increasing, therefore the computation cost is increased severely.

Based on these observation, we follow the minimal set principle, i.e. select minimal number of points needed to determine the model to be found.

3.3 Instantiate Model from Minimal Set of Points

The minimal number of points needed to instantiate a model is equal to the number of free coefficients in the model representation of the feature to be detected. For example, it is 2 for straight line, 3 for circle and 4 for ellipse.

Given a minimal set of points, the model is instantiated by solving the unknown coefficients in the equations for the model. Suppose that (x_1,y_1),...,(x_5, y_5) is selected as minimal set, then (x_i, y_i) is on the conic and we have:

$$A_i X = 0 \tag{3}$$

with

$$A_i = \begin{bmatrix} x_i^2 & x_i y_i & y_i^2 & x_i & y_i & 1 \end{bmatrix} \tag{4}$$

$$X = \begin{bmatrix} a & b & c & d & e & f \end{bmatrix}^T \tag{5}$$

Stacking equations from each point $(x_i, y_i), i = 1, ..., 5$ in to one set of equations, we get:

$$AX = 0 \tag{6}$$

$$A = \begin{bmatrix} A_1^T \cdots A_5^T \end{bmatrix}^T \tag{7}$$

the unknown $X = (a, b, c, d, e, f)^T$ is a homogeneous vector and has only 5 degrees of freedom, so, it can be solved from the 5 equations in Eq.(6). Since Eq.(6) is a set of homogeneous equations and the obvious solution $X = 0$ is meaningless, it can be solved by putting an additional conditional on the norm of

the unknown vector, e.g. $\|X\| = 1$. Instead, we turn Eq.(6) into a inhomogeneous set of equations by imposing a condition $X_i = 1$ for one unknown and some other conditions on the other unknowns. For example, in the case of a circle, a in Eq.(1) is sure to be nonzero, therefore the additional condition $X_1 = a = 1$ can be imposed. Further, $b = 0$ and $c = a$ can also be imposed for a circle. The number of free unknowns is left to be only 3. Based on these conditions, Eq.(8) can be derived from Eq.(6), it have 3 linear equations and 3 unknowns, the unknowns can be solved easily.

$$\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix} \begin{bmatrix} d \\ e \\ f \end{bmatrix} = \begin{bmatrix} -(x_1^2 + y_1^2) \\ -(x_2^2 + y_2^2) \\ -(x_3^2 + y_3^2) \end{bmatrix}. \tag{8}$$

3.4 Determine the Number of Samples Adaptively

The proportion of inliers ϵ is often unknown because we do not have statistics of foreground pixels and features in advance. Further more, ϵ is different with respective to different features and is varying while the detection process proceeds. Therefore, T_t can not be determined in advance.

We apply an adaptive strategy to solve this problem, i.e. determine ϵ and T_t adaptively while detection proceeds. It records the maximal value of ϵ and use it to determine the necessary number of trials. The adaptive algorithm is as follows:

1. Let $T_t = \infty$, $\epsilon = 0$, $\epsilon_{max} = 0$ and set $C_{sample} = 0$.
2. While $T_t > C_{sample}$ do 3-7:
3. Sampling stage,
4. Voting stage,
5. Let $\epsilon = N_{inlier}/N_{total}$, $\epsilon_{max} = max(\epsilon_{max}, \epsilon)$,
6. Compute T_t from ϵ_{max} using Eq.(2),
7. Let $C_{sample} = C_{sample} + 1$.

where ϵ_{max} records the maximal value of ϵ, N_{inlier} is the number of inlier points found in each trial while as N_{total} is number of all points.

3.5 Voting Without Explicit Distance Computation

As shown in subsection 3.1, in the voting stage of RANSAC, it needs to determine the consensus set and this needs distinguishing points within a distance threshold T_d of the model from other points. Obviously, it needs distance computation and comparison which consume much time, and this is what previous method really do. In this section, we will show how the distance computation can be avoided and present a new voting method without involving explicit distance computation.

In fact, the voting stage needs only counting points within a region we called *inlier region* which centers at the model and dilates from it with diameter T_d.

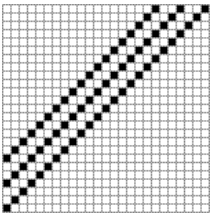

Fig. 3. Inlier region of a model: the center line indicates a model, the region between the up and below line indicates the inlier region and it contains only limited pixels

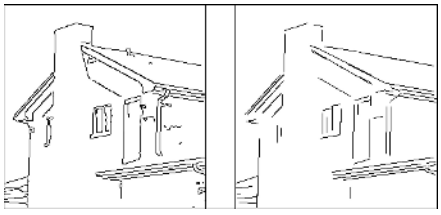

Fig. 4. Left image is an edge image of a house, and right image shows the detected line segments by our method

To do this, previous methods check all points by computing their distance from the model and count the ones whose distance smaller than T_d. But as shown in Fig.3, images contain only discrete pixels, and there are limited pixels in the inlier region. All the pixels in the inlier region are either foreground or background pixel, and only foreground pixels in the inlier region vote the model. Therefore, the alternative is checking all the pixels in the inlier region, if the pixel is foreground, it votes the model.

Now, let's assume that the model to be fitted is straight line. As shown in Fig.3, the inlier region of the line model is the region between two lines deviate from the model with a distance T_d. This region is equivalent to a line with width $2T_d$ centered at the model. This is also true for other models, therefore we have:

The inlier region of a model M is equivalent to a model M_e with width $2T_d$ centered at M.

Generating a line or curve with a width is a standard rasteration problem in computer graphics. Bresenham algorithm [12] is a widely used algorithm for rasteration, it can be implemented with only integer calculations and is fast. There are Bresenham algorithms [14] designed to generate straight line, circle and ellipse etc.

In this way, the distance computation and comparison is avoided, and this saves much computation time as will be shown in section 5. Furthermore, there is another advantage for applying rasteration method as an alterative to distance computation as shown in next subsection.

3.6 Explore Spatial Information in the Voting Stage

In fact, series of pixels are generated pixel by pixel in the rasteration methods. As shown in Fig.3, for example, Bresenham algorithm generate pixels from left to right, the successive pixels are spatially connected. Apparently, it is easy to record the consecutive foreground pixels and consecutive background pixels. In this way, we can detect spatially linked segments of straight line or curve. To account for noisy pixels in the original data and errors in edge detection, it should allow small gaps between segments. We set a threshold T_g for the gap between segments, segments with gap smaller than T_g are merged to be one

segment. Therefore, it does not need post-processing which is needed by RLD, RCD and most Hough Transform method.

4 Robust Model Fitting Based Feature Detection Algorithm

In this section, we present the proposed feature detection algorithm as follows:

1. Collect all foreground pixels in image I into set S and let N_{total} be the size of S,
2. Let $T_t = \infty$, $\epsilon = 0$, $\epsilon_{max} = 0$, $C_{sample} = 0$,
3. While $T_t > C_{sample}$ do 4-10:
4. Randomly select s points from S,
5. Determine a model M from the selected s points using method described in subsection3.3,
6. Apply Bresenham algorithm to generate pixels inside the inlier region of model M,
7. Count the number of spatially-linked foreground pixels of the generated pixels as N_{inlier},
8. Let $\epsilon = N_{inlier}/N_{total}$, $\epsilon_{max} = max(\epsilon_{max}, \epsilon)$,
9. Compute T_t from ϵ_{max} using Eq.(2),
10. Let $C_{sample} = C_{sample} + 1$,
11. If $N_{inlier} > T_{inlier}$ do 12-15:
12. Report the detected feature,
13. Remove the pixels on the detected feature from image I and corresponding data points from set S,
14. Let $N_{total} = N_{total} - N_{inlier}$,
15. go back to 2
16. Terminate.

where, T_{inlier} is a preset threshold for the minimal number of pixels one feature should have.

5 Experiments and Comparison

Based on section 4, we develop a Robust Model Fitting Based Line Detection method (RMFBLD) and apply it to both synthetic and real images to test its correctness and efficiency. Size of synthetic images is 256×256. The number of line segments in one image is used to control complexity of image. It ranges from 10 to 50 using 10 as step. Noise level is characterized by number of noise pixels. It ranges from 0 to 500 by a step of 50. For every level, 32 images are synthesized using different random seed. PPHT, RLD and RMFBLD are applied to detect line segments in these images. The total number of detected line segments and time used are shown in Tab. 2. As shown, RMFBLD is the most efficient method. Fig.5 shows one example of the results, it has 30 line segments and 300 noisy

Table 1. The necessary number of samples T_t for a given s and e

s	e				
	90%	80%	70%	60%	50%
2	3	5	7	11	17
3	4	7	11	19	35
4	5	9	17	34	72
5	6	12	26	57	146
6	7	16	37	97	293

Table 2. Comparison of RMFBLD with RLD and PPHT Method

Method	Detected line segments	Time (second)	Lines per second
RLD	3494	52,793	66.2
PPHT	18127	503,222	36.0
RMFBLD	18769	127,357	147.4

pixels, the original and detected line segments using RMFBLD, PPHT and RLD are shown from left to right, as shown, RMFBLD can detect features from image contains multiple features. As can be seen in both Tab. 2 and Fig.5, TRMFBLD and PPHT detect approximately the same number of lines, but RLD detects less lines.

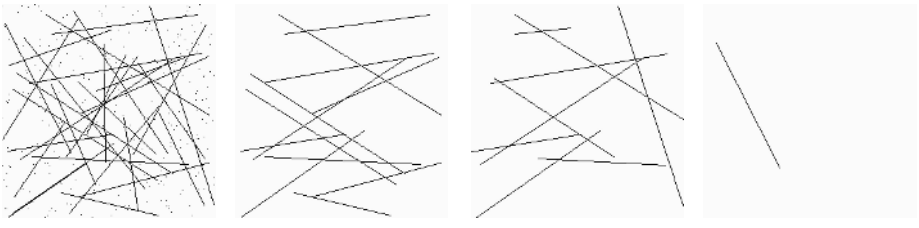

Fig. 5. Result example

Fig.4 shows another example, left figure is an edge image of an image of a house, and right one shows detected line segments by RMFBLD. It can be seen that global line features are successfully detected.

6 Conclusion

We formulate feature detection as robust model fitting problem. First, we use parameterized model to describe image features, and treat feature detection as model fitting problem. The global information is implicated in the parameterized model, global features can be easily detected without involving parameter transformation. Second, we adopt RANSAC as a solution to the model fitting problem. Because RANSAC is immune to outliers, the proposed method can deal with images contains multiple features and noisy pixels. Third, we develop a novel voting method for RANSAC, it avoids explicit distance computation by generating inlier pixels and checking if they are foreground. Besides the efficiency improvement, it provide a good chance to detect spatially connected feature.

Apart from presenting the framework of robust model fitting based image feature detection, we also develop Robust Model Fitting Based Line Detection

method for line detection at present. We plan to develop another method for detecting other features, such as circle and ellipse in the near future.

Acknowledgments

It is supported by the National Natural Science Foundation of China under Grant Nos.51476070101JW0409; the National Grand Fundamental Research 973 Program of China under Grant No.2002CB312101. The authors would like to thank the anonymous reviewers for very useful comments.

References

1. Boldt, M., Weiss, R., Riseman, E.: Token-based extraction of straight lines. IEEE Trans. System, Man, Cybernet **19** (1989) 1581–1594
2. Nacken, P.: A metric for line segments. IEEE TPAMI **15** (1993) 1312–1318
3. Illingworth, J., Kittler, J.: Survey: Survey of the hough transforms. Computer Vision, Graphics, and Image Processing **44** (1988) 87–116
4. Leavers, V.: Survey: Which hough transform. Computer Vision, Graphics, and Image Processing: Image Understanding **58** (1993) 250–264
5. Kiryati, N., Eldar, Y., Bruckstein, A.: A probabilistic hough transform. Pattern Recognition **24** (1991) 303–316
6. Xu, L., Oja, E., Kultanan, P.: A new curve detection method: Randomized hough transforms (rht). Pattern Recognition Letters **11** (1990) 331–338
7. Matas, J., Galambos, C., Kittler, J.: Progressive probabilistic hough transform. In: Proc. British Machine Vision Conference. (1998)
8. Chen, T., Chung, K.: A new randomized algorithm for detecting lines. Real-Time Imaging **7** (2001) 473–481
9. Chen, T., Chung, K.: An efficient randomized algorithm for detecting circles. Computer Vision and Image Understanding **83** (2001) 172–191
10. Zhang, Z.: Parameter estimation techniques: A tutorial with application to conic fitting. Image and Vision Computing **15** (1997) 59–76
11. Fischler, M., Bolles, R.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. ACM Commun. Assoc. Comp. Mach. **24** (1981) 381–395
12. Bresenham, J.: Algorithm for computer control of a digital plotter. IBM System Journal **4** (1965) 25–30
13. Rousseeuw, P.J.: Robust Regression and Outlier Detection. Wiley, New York (1987)
14. Rogers, D.F.: Procedural Elements for Computer Graphics. McGraw-Hill (1998)