A local edge detector used for finding corners

Fei Shen Han Wang
School of Electrical and Electronic Engineering
Nanyang Technological University, Singapore 639798
p146463962@ntu.edu.sg hw@ntu.edu.sg

Abstract

Many corner detectors make use of edge detectors to extract edge points first. In this paper, we proposed a local edge detection algorithm that can be used well in corner detectors. Instead of marking edge points globally, we detect them locally based on both gradient magnitude threshold and gray level analysis. The process is dynamic and can provide correct edge points for detecting corners. The problem of finding corners is simplified into detecting simple lines (straight lines that pass through the coordinate origin) in a local coordinate system. Hough transform is modified to organize the edge points generated by the new edge detector into simple lines. Experiments proved that our new corner detector based on the local edge detector works well over most images, and is fast enough for real time applications.

1 Introduction

As a key problem in image processing and computer vision, corner detection has drawn a lot of attention in the past twenty years. Many corner detectors have been reported. Zuniga and Haralick fit a continuous surface over a small neighborhood of each point and consider the rate of change in gradient direction [9]. Kitchen and Rosenfeld proposed a cornerness measure based on the change of gradient direction along an edge contour multiplied by the local gradient magnitude [7]. Wang and Brady observed that the total curvature of the gray level image is proportional to the second order directional derivative in the direction to edge normal and inversely proportional to the edge strength [5]. Moravec defined "points of interest" as points where there is a large intensity variation in every direction [4]. Harris and Stephens used image derivatives to estimate the autocorrelation of the image [1]. Zheng and Wang modified Plessey corner detector into the gradient direction corner detection, based on the measure of the gradient module of the image gradient direction and the constraints of the false corner response suppression [13]. Rangarajan, Shah and Brackle found an optimal function representing the corner detector which when convolved with the gray level function yields a maximum at the corner point [6]. Trajkovic and Hedley implemented the straightforward property of corners that the change of image intensity should be high in all directions [8]. Davies detected corners based on the generalized Hough Transform [3]. Xie $etal.$ devised a cost function to capture different desirable characteristics of corners and treated the corner detection as a problem of cost optimization [12]. Smith and Brady proposed a novel corner detection algorithm known as SUSAN corner detector based on brightness comparison [11]. A detailed analysis of these corner detectors can be found in [13].

Most popular corner detectors work on gray level image directly. They don't need to extract edge points firstly before detecting corners, which leads to the ambiguous structure of corner points. The corner detectors that depend on only boundary analysis suffers from the errors of previous edge points segmentation. General edge detector cannot localize edge points well around corners because of the rounding effect, leading to errors in reporting corners. In this paper, we proposed a local edge detection algorithm that can be used well in corner detectors. Instead of marking edge points globally, we detect them locally based on both gradient magnitude threshold and gray level analysis. The process is dynamic and can provide correct edge points for detecting corners. A new corner detector based on this local edge detection scheme is developed. The problem of finding corners is simplified into detecting simple lines (straight lines that pass through the coordinate origin) in a local coordinate system. Hough transform is modified to organize the edge points generated by the new edge detector into simple lines. Experiments proved that our new corner detector based on the local edge detector works well over most images, and is fast enough for real time applications.

In part 2, we will introduce the local edge detector. A new corner detector based on it is described in part 3. Comparative study of the results obtained with our corner detector and other corner detectors are carried out in part 4. Part 5 is the conclusion.

2 Detecting Edge Points Locally

The image surface $S(x, y)$ is described by the equation:

$$S(x, y) = x\overrightarrow{i} + y\overrightarrow{j} + I(x, y)\overrightarrow{k} \tag{1}$$

where $I(x, y)$ is the gray level value. Let I_x and I_y approximate the partial derivatives of the image. The gradient of intensity at any point (x_0, y_0) is defined as:

$$\nabla I(x_0, y_0) = (\frac{\partial I}{\partial x}(x_0, y_0), \frac{\partial I}{\partial y}(x_0, y_0)) \tag{2}$$

In the following part, using "edge point", we mean such a pixel, under certain edge measurement, whose edge response is large enough. Using "straight line", we mean a straight line in an image formed by some edge points. Using "simple

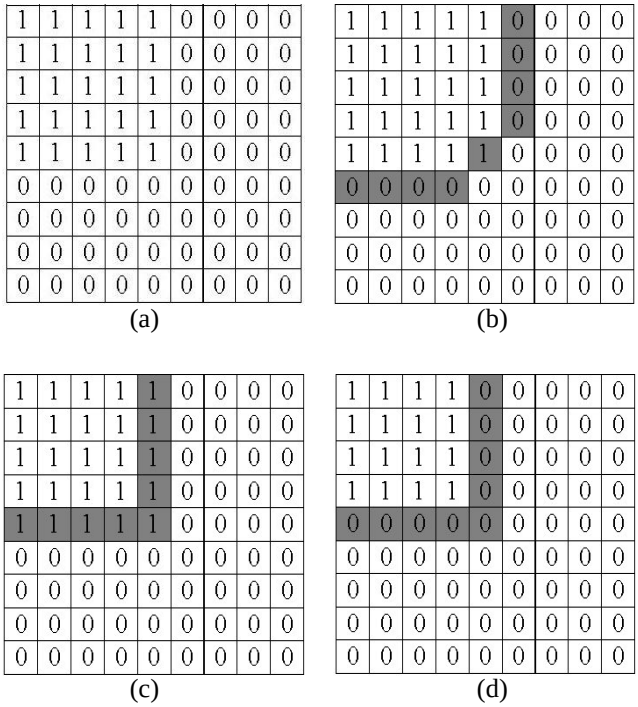


Figure 1: Detecting edge points dynamically

line", we mean a straight line that passes through the coordinate origin.

We argue that the edge points generated by common edge detectors are not suitable for corner detection. The reasons can be generalized as below. Firstly, for edge detectors that depend on Gaussian smoothing, there is obviously rounding effect at corner neighborhood, which leads to the ill localization of corner position. Secondly, the nonmaximum suppression used in common edge detectors will make the straight lines curved. This effect can be seen from Fig.1. Fig. 1a is an ideal binary L-Junction example. The corner region and the background region are marked by $'1'$ and $'0'$ individually. The central pixel is the exact corner position, with two simple lines passing through it. Fig. 1b shows the ordinary edge detection result of the previous figure. Edge points are highlighted as gray pixels. These edge points are more likely to form one curved line than two straight lines, which is not desirable.

To overcome the problems mentioned above, we detect the edge points window by window locally instead of finding them globally. Assume that a window of certain size is moved pixel by pixel in the image. The central pixel of the window is regarded as a corner candidate. All the pixels inside the window constitute a region of interest. The edge response of each pixel in the window is given by its gradient magnitude: $|\nabla I|$. To make sure that the lines are only one pixel wide, we need a thinning algorithm, but nonmaximum suppression is not qualified based on above analysis. Remember that the corner point is the intersection of two or more straight lines. So it's natural for us to require that the edge points that form the simple lines and the corner point belong to same sub region, which means their gray level values are similar. This constraint not only thins the edge points effectively, but also reduces the noise effect strongly, taking the place of Gaussian smoothing.

For any pixel (x, y) in the region of interest, if the following two equations are satisfied, it can be marked as an edge point.

$$|\nabla I(x,y)| > t_1 \tag{3}$$

$$|I(x,y) - I(0,0)| < t_2 \tag{4}$$

$I(0,0)$ is the gray level value of the central pixel and t_1, t_2 are the thresholds to be determined. Obviously, the detected edge points are subjective to the window on which we are currently working. So our edge detection scheme is a dynamic process. The result of our edge detection scheme applied to Fig. 1a is shown in Fig. 1c. As the gray level value of central pixel is $'1'$, the edge points whose gray level values are $'0'$ are cleared off. If the window center moves to the corner point marked by $'0'$, the edge points change to pixels whose edge responses are large and gray level values are $'0'$, as shown in Fig. 1d. At both situations, the detected edge points fit our requirement.

3 A New Corner Detector

We move a window with a suitable size pixel by pixel to detect corners. Each window is treated as a local coordinate system, with the central pixel being the origin. If the central pixel is a corner point, the following two conditions should be satisfied.

- Under certain edge measurement, its edge response should be large enough, as corner point should also be an edge point.
- We should be able to detect at least two straight lines passing through it, that is two simple lines should exist.

Assume we have got a set of edge points by the local edge detector in a certain window. Hough transform is modified to organize these points into simple lines, whose parameter equation is:

$$xsin\theta + ycos\theta = 0 \tag{5}$$

There is only one parameter θ in this equation. Regarding (x, y) as fixed, the equation corresponds to one value $\theta = arctg(-\frac{y}{x})$ in the θ space, or parameter space. All edge points on the same simple line will correspond to a same θ in parameter space. As points that even lie on a non-simple line contribute to different θ values, the modified Hough transform reports simple lines steadily.

Line extraction with ordinary Hough transform has a computational complexity $O(n * m)$, where n is the number of edge points and m is the size of a discretization of the parameter space. Reducing the size of parameter dimension can speed Hough transform significantly. Our implementation of Hough transform is only one dimensional, which makes our corner detector fast enough for real time applications.

4 Experimental Results

In this part, we compare the performance of our new corner detector with other three widely-used corner detectors, which are Wang and Brady corner detector (WANG-BRADY), Plessey corner detector (PLESSEY) and SUSAN corner detector (SUSAN). Experiments are performed on synthetic image and real images.

First, we compare the performance of these four corner detectors on a synthetic image. This synthetic image consists of many corner types, including L-Junction, Y-Junction, T-Junction, Arrow-Junction and X-Junction. It has been widely used to evaluate how accurately an corner detector responds to different corner types. Fig. 2a was its corner map obtained by our detector. Fig. 2b to 2d showed the corner images produced by WANG-BRADY, PLESSEY and SUSAN individually. From the result, we can see that all the corners in this image are correctly reported by our corner detector. No wrong corners were reported. Wang-Brady corner detector missed one corner, at the obtuse angle of the triangle. Also, Wang-Brady corner detector spotted spurious corners when the angles are very sharp. This shows the disability of WANG-BRADY of detecting corners whose angle is very large or very small. Although Plessey corner detector reported all the corners, it marked three corners wrongly. What's more, Plessey corner detector suffers great deviation in corner localization. In the T-Junctions at the low part of the ladder, the localization deviation is even as large as 4 pixels. SUSAN performs quite well too. But it missed the X-Junction in the middle of the square at the bottom of the synthetic image. In this X-Junction, the gray level of the upper-left region equals to that of the downright region and the gray level of the upper-right region equals to that of the down-left region. In this situation, the response of the central pixel of SUSAN will look like an edge point, instead of a corner point. This shows that SUSAN cannot work well at some X-Junctions.

Also we tested the corner detectors on a real image. This real image is a lab scene There are many rectangles with well-defined corners in the image. The boundaries of these rectangles are smoothed in a certain degree. We can test the ability of these detectors to detect smooth corners based on this image. The corner maps are shown as Fig. 3a to 3d. Our algorithm and Plessey corner detector performed best in this test image. Almost all the corners were detected successfully. WANG-BRADY missed some corners that are corrupted by noise and reported some noisy points. Although SUSAN is less sensitive to noise, it missed many smoothed corners, which shows it can not work well on highly smoothed images.

Speed is an important requirement for real-time jobs such as robot navigation. Our algorithm is based on 1D Hough transform, which determines the low algorithm complexity. To give an explicit explanation, the algorithms were run on several $256 * 256$ size images for hundreds of times on a Pentium III 500 PC and the average running time was shown below. Our algorithm delivers a speed performance of $180ms$ per frame. WANG-BRADY takes about $110ms$. SUSAN has the best speed performance, using about $45ms$. PLESSEY is the slowest one among these algorithms and takes about $538ms$ per frame. Although our algorithm is a little slower than WANG-BRADY and SUSAN, it is 3 times faster than PLESSEY. This experiment shows the ability of implementing our algorithm in real-time applications.

5 Conclusion

We have introduced a local edge detection algorithm that can be well used for detecting corners. Instead of marking edge points globally, we detect them locally based on both gradient magnitude threshold and gray level analysis. The process is dynamic and can provide correct edge points for detecting corners. The problem of finding corners is simplified into detecting simple lines (straight lines that pass through the coordinate origin) in a local coordinate system. Hough transform is modified to organize the edge points generated by the new edge detector into simple lines. Experiments proved that our new corner detector based on the local edge detector works well over most images, and is fast enough for real time applications.

References

[1] C. Harris and M. Stephens, "A combined corner and edge detector", *Proc. 4_{th} Alvey Vision Conference*, pp. 189-192, 1988.

[2] D.H. Ballard and C.M. Brown, "Computer Vision", *Prentice-Hall*, 1982.

[3] E.R. Davies, "Application of the generalised Hough Transform to corner detection", *IEE Proceedings*, Vol. 135, pp. 49-54, 1988.

[4] H.P. Moravec, "Towards atiomatic visual obstacle avoidance", *Proc. 5_{th} International Joint Conference on Artificial Intelligent"*, pp. 584, 1977.

[5] H. Wang and M. Brady, "Real-time corner detection algorithm for motion estimation", *Image and Vision Computing*, Vol. 13:9, pp. 695-703, 1995.

[6] K. Rangarajan, M. Shah and D.V. Brackle, "Optimal corner detector", *Comouting Vision, Graphics, and Image Processing*, Vol. 48, pp. 230-245, 1989.

[7] L. Kitchen and A. Rosenfeld, "Gray level corner detector", *Pattern Recognition Letters*, Vol. 1, pp. 95-102, 1982.

[8] M. Trajkovic and M. Hedley, "Fast corner detector", *Image and Vision Computing*, Vol. 16, pp. 75-87, 1998.

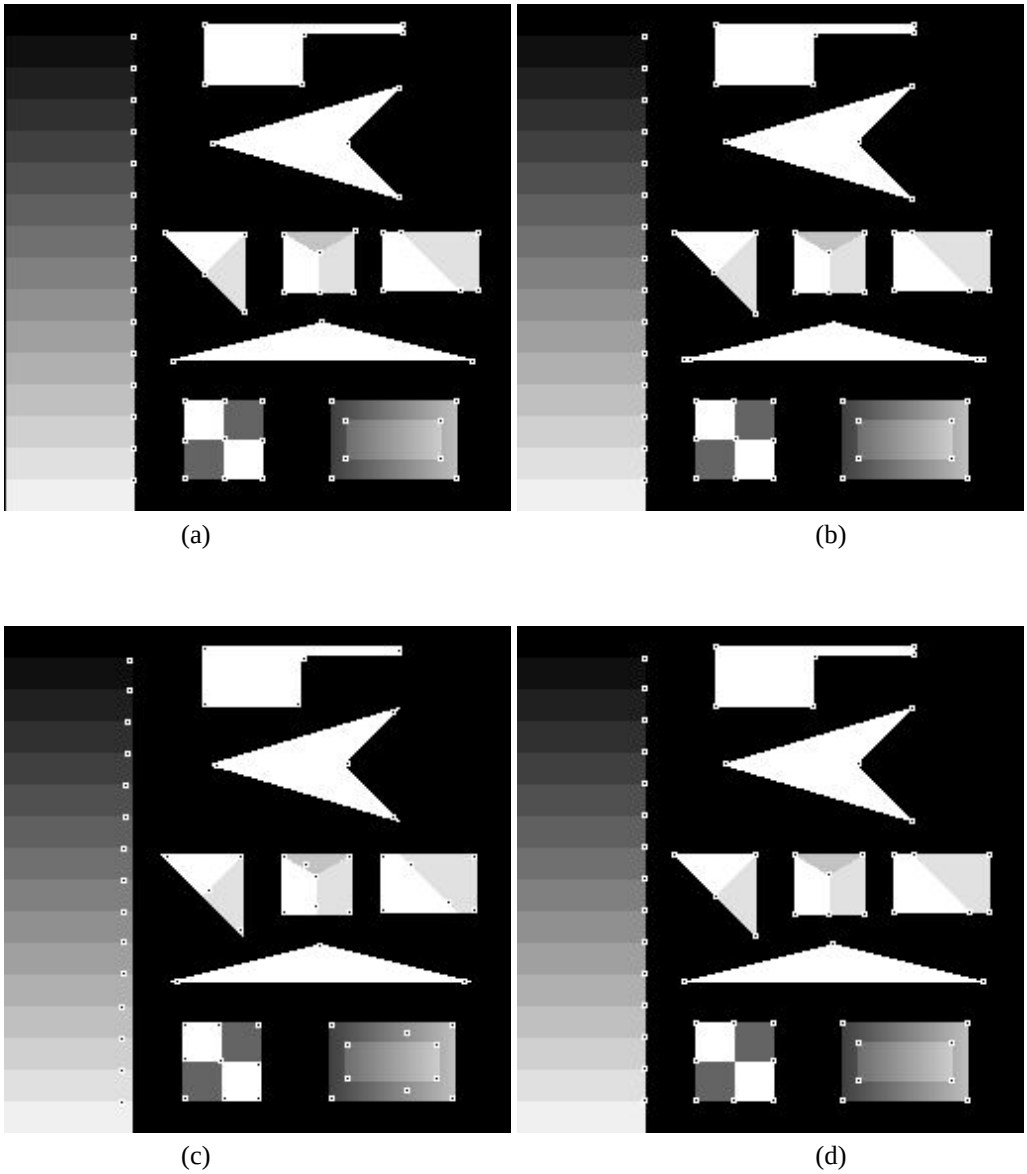
[9] O.A. Zuniga and R.M. Haralick, "Corner detection using facet model", *Proc. Conference on Pattern Recognition and Image Processing*, pp. 30-37, 1983.

[10] P.R. Beaudet, "Rotaional invariant image operators", *Proc. 4_{th} International Conference on Pattern Recognition*, pp. 579-583, 1978.

[11] S.M. Smith and M. Brady, "SUSAN - a new approach to low level image processing", *International Journal of Computer Vision"*, Vol. 23(1), 45-78, 1997.

[12] X. Xie, R. Sudhakar and H. Zhuang, "Corner detection by a cost minimization approach", *Pattern Recognition*, Vol. 26, No. 8, pp. 1235-1243, 1993

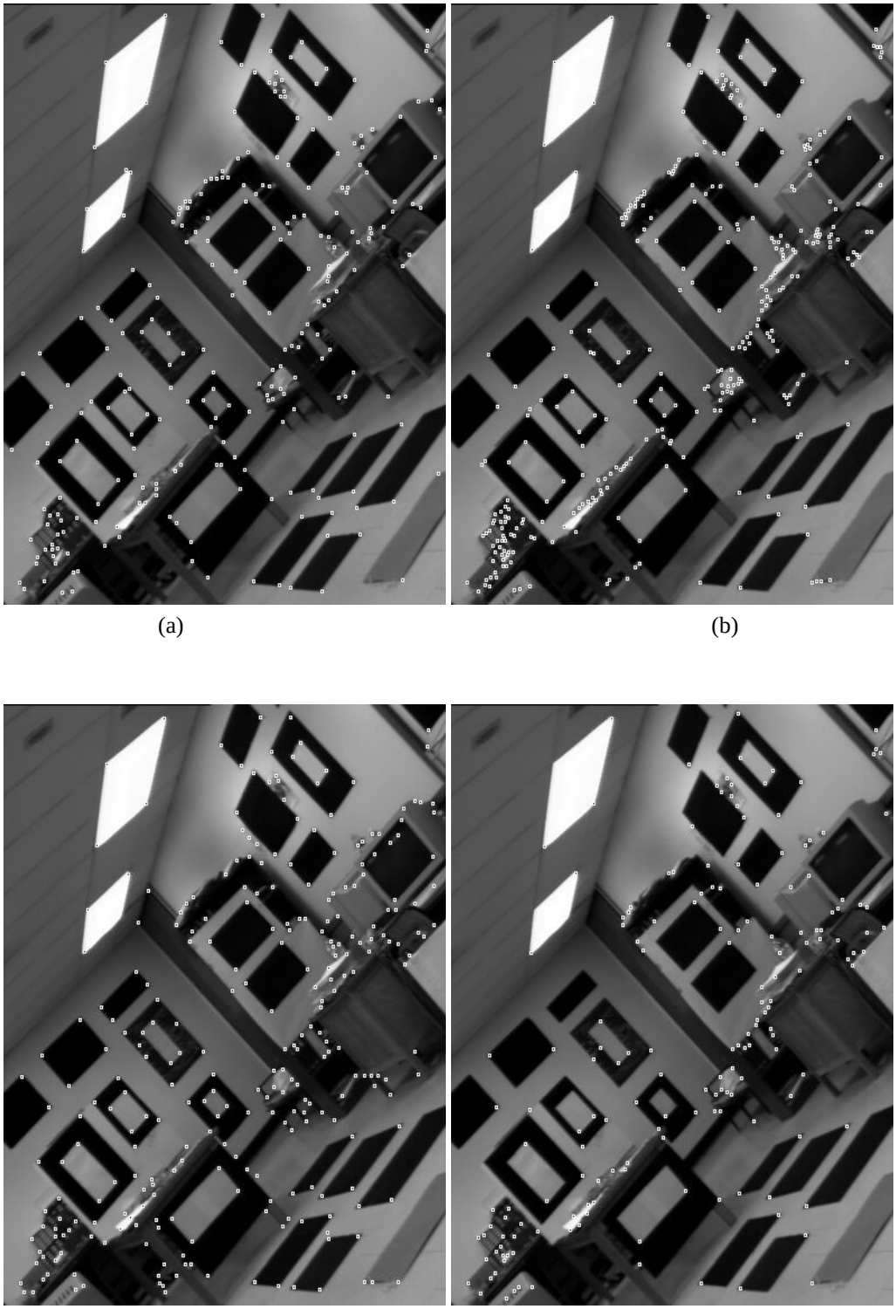
[13] Z. Zheng, H. Wang and E.K. Teoh, "Analysis of gray level corner detection", *Pattern Recognition Letters*, Vol. 20, pp. 149-162, 1999.



(a) (b)

(c) (d)

Figure 2: Corner maps of the synthetic image obtained by: (a) New Algorithm; (b) Wang and Brady Corner Detector; (c) Plessey Corner Detector; (d) SUSAN Corner Detector



(a) (b)

(c) (d)

Figure 3: Corner maps of the lab indoor image obtained by: (a) New Algorithm; (b) Wang and Brady Corner Detector; (c) Plessey Corner Detector; (d) SUSAN Corner Detector