# An Improved Canny Edge Detection Algorithm Based on DCT

Miaomiao Zhao, Hongxia Liu and Yi Wan
School of Information Science and Engineering
Lanzhou University, China, 730000
Email: zhaomm13@lzu.edu.cn, ...

***Abstract*—The Canny edge detection algorithm is one of the most widely used in many computer vision and image analysis applications. However, it also tends to be sensitive to noise so that some the weak edge information can be easily lost when using the Gaussian filter to smooth images. In this paper, we propose a new method that replaces the traditional Gaussian filtering step in the Canny algorithm. In this method, the DCT coefficients are nonlinearly shrinked to achieve the denoising effect. Experiments show that the proposed methold produces better edge detection results at preserving more valid edge information and that it is more robust to noise than the traditional Canny edge detection algorithm.**

***Index Terms*—Edge detection, DCT coefficients, Image denoising, Canny operator.**

## I. Introduction

Edge detection is an important part in image processing and computer vision. The purpose of edge detection is to identify the change of brightness including large parts of the image feature information and shape quality. As is well-known, Many classical edge detection operators have been proposed, such as Robert, Prewitt, Kirsch, Laplace, LoG, Sobel and Canny(e.g, [1]). Different operators have different advantages. Because the better signal to noise ratio and the detection accuracy it has, the Canny Operator has been the benchmark for evaluating other metholds (e.g., [2], [3]).

Besides some well-known classical edge detection algorithms, some improved edge detection algorithms based on Canny operator have been put forward. Recently, Weibin Rong proposed that the gravitational field intensity can replace traditioal Canny image gradient (e.g, [4]). In [5] the bilateral filtering based edge detection was presented which generated well localized edges, but the noise removal capability of this methold is not very strong. Xiangdan Hou introduced an improved Canny algorithm based on the histogram-based fuzzy c-means clustering algorithm (e.g, [6]). It showed good edge detection results to some extend.

However, the image quality is easily influenced by some factors, especially the noises. The traditional Canny edge detection methold cannot adapt to different noise situations. Therefore, this paper provide a new methold to improve the edge detection quality, this methold is more effective at preserving more valid edges because it does a better work for denoising.

The rest of the paper is organized as follows. In Section II we present the traditional Canny algorithm. In Section III we present the improved canny algorithm. In Section IV we give the experimental results. Then, we draw conclusions in Section V.

## II. The Traditional Canny Algorithm

The traditional Canny algorithm is widely used for practical image edge detection. The usual way to detect edges in color images is to convert the color image into gray image first. In this section, we will give the details of the traditioanl Canny edge detection. The algorithm's process is:

Step 1: Color conversion
Step 2: Gaussian smoothing
Step 3: Calculation of the image gradient
Step 4: Non-maxima suppression of the grad value
Step 5: Edge determination and connection

### A. Color conversion

In this step, two commonly used methods to convert an RGB color image into a grayscale image $I$ are:

$$I = R * 0.299 + G * 0.587 + B * 0.114 \tag{1}$$

$$I = \frac{R + G + B}{3} \tag{2}$$

Eq.(1) is very popular in image processing, while as Eq.(2) of getting the average of three colors is also suitable.

### B. Gaussian smoothing

It is necessary to smooth images for removing the noise before edge detection. Using the Gaussian function is the first choice for classical Canny edge detection algorithm. Canny algorithm generally use 2-D Gaussian function to smooth image and denoise(e.g, [7]).

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{3}$$

The gradient vector is:

$$\nabla \mathbf{G} = \left[\frac{\partial G}{\partial x}, \frac{\partial G}{\partial y}\right]^T \tag{4}$$

The two-dimensional Gaussian filter can be divided into two row and column filters as:

$$\frac{\partial G}{\partial x} = k \cdot x \cdot \exp(-\frac{x^2}{2\sigma^2}) \cdot \exp(-\frac{y^2}{2\sigma^2}) \tag{5}$$

$$\frac{\partial G}{\partial y} = k\cdot y\cdot \exp(-\frac{y^2}{2\sigma^2})\cdot \exp(-\frac{x^2}{2\sigma^2}) \tag{6}$$

Where $\sigma$ stands for the standard deviation (size) of the Gaussian filter, which controls the smoothing degree, and $k$ is a constant. When $\sigma$ is small, the smoothing ability is weak, but the edge preservation ability is strong; Whereas, when $\sigma$ is big, the smoothing ability is strong, but the edge preservation ability will be weak.

### *C. Calculation of the image gradient*

To calculate the image gradient, the traditional Canny edge detection algorithm selects a $2 \times 2$ neighboring area to get the magnitude and direction the gradient(e.g, [4], [8]). The first order derivative on $X$ and $Y$ directions can be get from following formulas:

$$E_x(i,j) = \frac{1}{2}(I(i+1,j) - I(i,j) + I(i+1,j+1) - I(i,j+1)) \tag{7}$$

$$E_y(i,j) = \frac{1}{2}(I(i,j+1) - I(i,j) + I(i+1,j+1) - I(i+1,j)) \tag{8}$$

Therefore, we can get the template of the canny operator:

$$G(x) = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}, \quad G(y) = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \tag{9}$$

The image gradient magnitude is :

$$M(i,j) = \sqrt{E_x(i,j)^2 + E_y(i,j)^2} \tag{10}$$

The image gradient direction is:

$$\Theta(i,j) = \arctan\frac{E_y(i,j)}{E_x(i,j)} \tag{11}$$

### *D. Non-maxima suppression of the grad value*

Canny method use the $3 \times 3$ neighboring area to compare a pixel with its two adjacent pixels along the gradient direction. If its $M(i,j)$ is bigger than the two adjacent have, the pixel will be marked as a candidate edge point, and its $M(i,j)$ keeps unchanged; else $M(i,j) = 0$.

### *E. Edge determination and connection*

As [4] describes, the Canny algorithm chooses a double-threshold to select edge points after non-maxima suppresion of the gradient. Usually, the higher threshold is $2 \sim 3$ times the lower threshold. Thus, the pixels which have higher gradient magnitudes than the higher threshold will be marked as edge points, and the rest will be marked as candidate edge points. Those candidate edge points which are connected with edge points will be marked as edge points. Now, the edge image is obtained. This step can reduce the influence of noise on the edge selection of the final edge image.

### *F. Disadvantages of the traditional Canny edge detection algorithm*

The traditional Canny edge detection usually uses the Gaussian function to smooth image, but it does not have very good performance in dashed areas, and some true edge lines are lost. For this reason, this paper introduces the concept of DCT compressed domain, converts the 2D discrete pixels to continuous DCT domain, through DCT transformation to get the DCT coefficients and estimating noises, to make corrections of DCT coefficient, then, we can obtain the smooth image by IDCT.

## III. The Improved Canny Algorithm

Traditional Canny method uses Gaussian function to denoise. Generally speaking, the smooth degree always depend on the size of the Gaussian kernel, which is directly affects the quality of edge detection. Meanwhile, we find that the result of using Gaussian function to denoise is not very satisfactory, beacuse it is limited on removing the Gaussian noises. While the images are always affected by various noises such as Salt and Pepper noise, Rayleigh noise, Gaussian white noise. In order to resolve these problems, we propose an improved Canny edge detection algorithm based on DCT. As we all know, image function can be described by the DCT coefficient on the basis of IDCT. So we can use the DCT formula all through the image to get the DCT coefficients, and deal with those coefficients, then get a high quality image. The main algorithm as follow.

### *A. DCT transformation*

Suppose, the matrix of an original image $I(x,y)$ is $M \times N$, and it is maked up with series of discrete pixels. DCT transformation can make the discrete images into continuous domain, using sine and cosine functions. The DCT transformation (e.g, [9])as follow:

$$F(u,v) = c(u)c(v)\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} I(x,y)\cos\left[\frac{(x+0.5)\pi}{M}u\right]\cos\left[\frac{(y+0.5)\pi}{N}v\right] \tag{12}$$

$$c(u) = \begin{cases} \sqrt{\frac{1}{M}}, & u = 0 \\ \sqrt{\frac{2}{M}}, & u \neq 0 \end{cases} \tag{13}$$

$$c(v) = \begin{cases} \sqrt{\frac{1}{N}}, & v = 0 \\ \sqrt{\frac{2}{N}}, & v \neq 0 \end{cases} \tag{14}$$

Where $u,\ x = 0,1,2...M-1;\ v,\ y = 0,1,2...N-1.$

### *B. DCT coefficient correction*

From Eq.(12), we can get the DCT coefficients $F(u,v)$, the value of the coefficients represent image informations. The upper left corner are the low frequency components, whose coefficients are the target contour and gray level distribution. The lower right corner are the high frequency components, such as noises, which will have an influence on the edge extraction. So, we need to do something for DCT coefficients to denoise. This paper provides a formula to correct them, the formula is described as follow:

$$F^{'}(u,v) = \frac{F^3(u,v)}{F^2(u,v)+\xi} \tag{15}$$

Where the parameter $\xi$ is to be used for correction the original DCT coefficients $F(u,v)$. We can see, if the original DCT coefficient is relatively a large value, it just nothing change after coefficient correction, while, if the coefficient is relatively small, it will become much smaller. So this process can enhance the edge and denoising. Choosing suitable $\xi$ is very important for different images, we usually set $10 < \xi < 150$.

### C. IDCT transformation

Using IDCT transformation(e.g, [9]) for DCT coefficients to get the smooth image $I^{'}(x,y)$ .

$$I^{'}(x,y) = c(u)c(v)\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}$$

$$F^{'}(u,v)\cos\left[\frac{(x+0.5)\pi}{M}u\right]\cos\left[\frac{(y+0.5)\pi}{N}v\right] \tag{16}$$

Through DCT transformation, coefficients correction, and IDCT transformation, we can get a smooth gray image, then, do step 3 to step 5 for each pixel showed in Section II.

## IV. Experimental Results

In this section, we compare the performance of our proposed method with the traditional method. Twenty test images are used in this experiment, four of the test images are used to show the edge detection results. All the codes are written in $C++$ and run under the Linux system. We use Subjective Evaluation and Objective Evaluation to show their performance.

### A. Subjective Evaluation

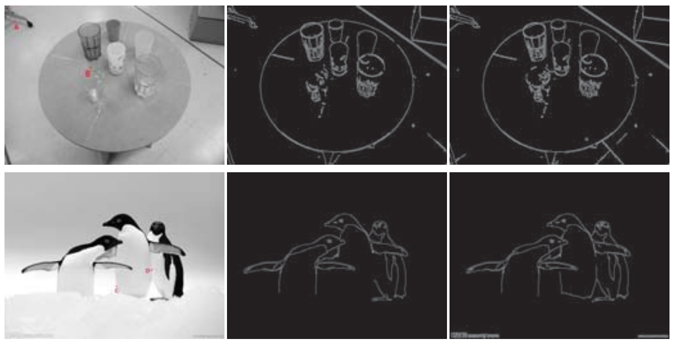

Fig. 1. Poor edge images. (a) Original images. (b) Traditional Canny methold. (c) Improved methold.

Fig. 1 shows the images with poor edges. The original images of desk and penguins in (a). In the first row, it is the desk image, we hope that it could obtaines more edge lines on the desk and floor. And the second row, it is the penguins image, we want to keep more continuous edge lines. Traditional Canny method could also detect the edges in (b), but the edges are not continuous and some lines cannot be detected. By comparison, we can see our proposed method in (c) can detect continuous edges, preserve more local edge informations and it also can detect more edge lines on the image's background. Especially, where are marked out with red color in Fig. 1, such as A, B, C and D.

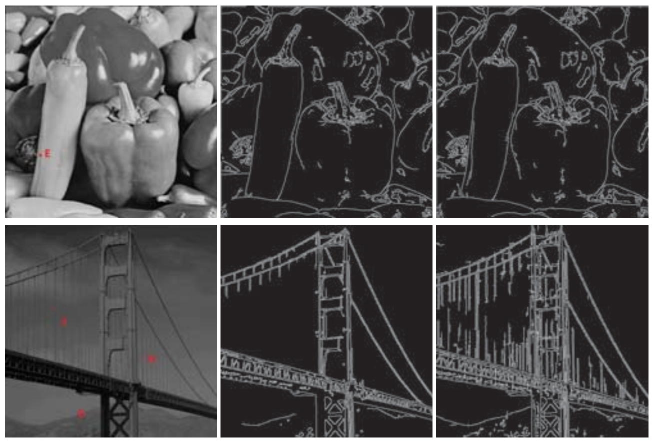

Fig. 2. Rich edge images. (a) Original images. (b) Traditional Canny methold. (c) Improved methold.

Fig. 2 shows the images with rich edges. The original images of peppers and goldgate. The difference between Fig. 2 and Fig. 1 is that there are more edge lines of the original image in Fig. 2. We aim at detecting more true edges use the traditional Canny method and our proposed method. From the peppers image and goldgate image, we can see that our proposed algorithm in (c) can preserve more edge lines and the slimsy lines of chains than the traditional method in (b). Especially, where are marked out in E, F, G, H with red color.

Above all, we can make a conclusion from Fig. 1, Fig. 2, our improved algorithm shows a better performance in preserving more local edge informations than the traditional method either on poor edge images or rich edge images.

### B. Objective Evaluation

Except for the subjective evaluation, the objective evatuation is also very significant. Reference [10] provides some edge detection evaluation criterions such as: The total edge point numbers, PSNR and Bit error. We take various images to evaluate it.

(1) The total edge point numbers

In order to compare the performance between the traditional method and our proposed method, we ensure that this two algorithms under the premise of the same high threshold and low threshold. In this condition, we respectively get the total edge point numbers of each images, which are obtained from the traditional method and proposed method. Generally

TABLE I
THE TOTAL EDGE POINT NUMBERS BETWEEN TRADITIONAL METHOLD AND PROPOSED METHOLD

| | Low-threshold | High-threshold | Traditional edge pixels | Proposed edge pixels | $\xi$ |
|---|---|---|---|---|---|
| Desk image | 33 | 89 | 9466 | 10495 | 30 |
| Penguins image | 40 | 120 | 6329 | 9513 | 60 |
| Peppers image | 55 | 125 | 14429 | 15277 | 135 |
| Goldgate image | 40 | 100 | 15772 | 36421 | 145 |

TABLE II
COMPARISON OF PSNR VALUE AND BIT-ERROR VALUE OF TRADITIONAL METHOLD AND PROPOSED METHOLD

| image | Traditional PSNR | Proposed PSNR | Traditional Bit-error | Proposed Bit-error |
|---|---|---|---|---|
| Desk image | 40.51 | 45.29 | 0.027 | 0.026 |
| Penguins image | 41.05 | 44.60 | 0.024 | 0.022 |
| Peppers image | 35.42 | 39.18 | 0.028 | 0.025 |
| Goldgate image | 36.81 | 37.35 | 0.025 | 0.022 |

speaking, the more edge point numbers means the better performance on edge detection.

(2) PSNR

PSNR is most widely used in evaluation method. When we respectively use the Gaussian function and DCT coefficient correction to get the smooth image, the smooth image usually looks rarely completely the same as original image. PSNR can be used to distinguish between high quality image and low quality image, it represents the ratio of siginal maximum power which could influence signal's accuracy. It can be calculated as follow.

$$PSNR = 10\log_{10}(\frac{255*255}{MSE}) \tag{17}$$

$$MSE = \frac{1}{MN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(I(i,j) - I^{'}(x,y))^2 \tag{18}$$

Where MSE is the mean square error between the original image and the processed image. $I(x,y)$ stands for an original image with the size of $M \times N$; $I^{'}(x,y)$ is the processed image after smoothing in the same size of the original image.

(3) Bit error

Bit error is the inverse of PSNR. The lower value of Bit error indicates better quality of a image.

$$Bit\ error = \frac{1}{PSNR} \tag{19}$$

Table I shows the total edge point numbers both the traditional Canny method and the proposed method. PSNR and Bit error of different images between traditional method and improved method are shown in Table II. We can see that our proposed method have much more edge point numbers, higher value of PSNR and more lower Bit error in the test images.

## V. CONCLUSIONS

An improved Canny edge detection algorithm is presented in this paper. Our method based on DCT transformation, handling the DCT coefficients and inversing DCT, which will replace the traditional Gaussian function for denoising. After that, we calculate the image gradient, non-maxima suppression and edge connection to get the final edge image. This paper gives the details about the traditional Canny edge detection steps and our proposed method approach. Comparison results show that our improved method outperforms the traditional Canny edge detection method for keeping more edge informations and enhancing the ability of noise supression.

## REFERENCES

[1] J Canny, "A Computational Approach to Edge Detection", IEEE Trans. on PAML, vol.8, no.6, pp.679-698, 1986

[2] F A Pellegrino, "Edge Detection Revisited", IEEE Trans On System Man and Cybernetics, vol.34, no.3, pp.251-261, 2011

[3] T B Nguyen, D Ziou, "Contextual and Non-contextual Performance Evaluation of Edge Detectors", Pattern Recognition Letters, vol.21 no.8, pp.805-816, 2000

[4] R W Bin, L Z Jing, Z Wei, S L Ning, "An Improved Canny Edge Detection Algorithm", Proceedings of 2014 IEEE International Conference on Mechatronics and Automation Aug. Tianjin, pp.577-582, 2014

[5] S Jahanzed, H Pirzada, A Siddiqui, "Analysis of Edge Detection Algorithms for Feature Extraction in Satellite Images", International Conference on Space Science and Communication Malaysia, IEEE, pp.238-242, 2013

[6] X D Hou, Y F Dong, H Zhang, J H Gu, "Application of A Self-adaptive Canny Algorithm for Detecting Road Surface Distress Image", 2009 Second International Conference on Intelligent Networks and Intelligent Systems, pp.354-357, 2009

[7] X Geng, K Chen, X G Hu, "An Improved Canny Edge Detection Algorithm for Color Image", 2012 10th IEEE International Conference on Industrial Informatics (INDIN), pp.113-117, 2012

[8] Y D Jia, "Machine Vision", Beijing: Science Press, pp.97-100, 2000

[9] Z Y Liu, X T Cui, X H Li, "Algorithm for Image Edge Detection Based on DCT Compressed Domain", Computer Technology and Development, vol.18, no.4, 2008

[10] Chinu, A Chhabra, "A Hybrid Approach for Color Based Image Edge Detection", Advances in Computing, Communications and Informatics (ICACCI), pp.2443-2448, 2014