IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

EDGE DETECTION ALGORITHMS ON DIGITAL SIGNAL PROCESSOR DM642

Dr. H S Prasantha (0000-0003-4739-517X)
Professor
Department of Electronics & Communication Engineering
Nitte Meenakshi Institute of Technology, Bangalore, Karnataka, India

Abstract: Image Segmentation plays an important role in the area of image processing with variety of applications in Medical, Remote sensing and agriculture to mention a few. A lot of research work is currently in progress with various related areas resulting in many computationally efficient algorithms. Edge detection is an important step in various fields such as pattern recognition, image segmentation, and scene analysis. The choice of the algorithm in most applications depends on the application and image in question rather than a generalized method. The paper focuses on a real-time implementation of edge detection techniques on gray and color images. The algorithms are implemented in real time using Digital Signal Processor (TI) TMS320DM642. The results show that the low computation time with the accurate edge results.

Index Terms **– Image Segmentation, edge detection, operator, computation time, Digital Signal Processor**

1. INTRODUCTION

Digital Image Processing finds an important role in variety of fields such as Medical fields, agriculture, military, and communication fields. The image processing can be used in variety of fields and applications. The applications of image processing include scaling, segmentation, compression, enhancement, detection, feature extraction, restoration etc. Segmentation of an image entails the division or separation of the image into regions of similar attribute. The most basic attribute for segmentation is image luminance amplitude for monochrome image and color components for the color image. Segmentation is required to distinguish objects from background.

Edge: Edges characterize boundaries and are therefore a problem of fundamental importance in image processing. The edge of an image is those points at which the luminous intensity changes sharply, which usually reflects important events and changes in properties of the world. In image processing, an edge is often interpreted as one class of singularities. In a function, singularities can be characterized easily as discontinuities where the gradient approaches infinity. However, image data is discrete, so edges in an image often are defined as the local maxima of the gradient. Generally, an edge is defined as the boundary pixels that connect two separate regions with changing image amplitude attributes such as different constant luminance and tristimulus values in an image.

Edge detection: Edge detection is a type of image segmentation techniques which determines the presence of an edge or line in an image and outlines them in an appropriate way. Edge detection is nothing but the process of identifying and locating sharp discontinuities in an image. The discontinuities are abrupt changes in pixel intensity which characterize boundaries of objects in a scene. An edge detector is basically a high pass filter that can be applied to extract the edge points in an image. Image Edge detection significantly reduces the amount of data and filters out unwanted information, while preserving the important structural properties in an image. The detection operation begins with the examination of the local discontinuity at each pixel element in an image. Amplitude, orientation, and location of a particular subarea in the image that is of interest are essentially important characteristics of possible edges. Based on these characteristics, the detector has to decide whether each of the examined pixels is an edge or not.

Edge Detection Applications: Edge detection is a main tool in pattern recognition, image segmentation, and scene analysis. Edge detection and extraction are of great importance to identifying and understanding the whole image and has been applied to many fields such as

- Finger edge extraction and finger edge comparison for person’s identification.
- Edge detection in human auditory cortex.
- Tumor detection in magnetic resonance
- Interface identification in multiphase flow etc.

Image segmentation is widely used in Computer Vision related applications and becomes increasingly prevalent in biomedical engineering. Image segmentation is an important process of extracting features or regions of interest from an acquired image for further intelligent computer analysis. Segmentation using computer vision finds multiple applications especially in the area of bio-medicine. The tissues, tumors or organs of interest need to be extracted from the images acquired using medical imaging techniques. The clear view of only the interested regions/organs will help the radiologist in earlier diagnosis and treatment. Segmentation is also used intensively in areas such as pattern recognition, remote sensing, metallurgy etc. There are number of literatures on image segmentation both semiautomatic and automatic.

Many segmentation algorithms using edge detection have proved to be successful. But not much of work is done in the area of realizing hardware for any branch of image processing. The detailed survey reveals image segmentation and edge detection algorithms have been implemented on FPGA, systolic array architecture. The survey reveals edge detection algorithms are implemented on DSP processor TMS320C6711 for real time applications.

In edge detection methods, the drawbacks are due to falls edge detection, missing true edges, producing thin or thick lines, and also due to noise. Hence there is a need to design an edge detector operator to produce Edge magnitude, Edge orientation, High detection rate and good localization. There are many ways to perform edge detection. However the majority of different methods may be grouped into Gradient based edge detection [first order derivative] and Laplacian based edge detection [second order derivative].

II. RESEARCH METHODOLOGY

The proposed methodology involves the discussion of classical edge detectors, different edge detection operators with descriptions and the details about the DSP board used for implementation.

Review of Classical Edge Detectors:

Classical edge detectors use a pre-defined group of edge patterns to match each image segments of a fixed size. 2-D discrete convolutions are used here to find the correlations between the pre-defined edge patterns and the sampled image segment.

$$(f * e)(x, y) = \sum_{i,j} f(i,j)e(x-i, y-j),$$

Where *f* is the image and *e* is the edge pattern defined by

e(−1, −1)	e(−1, 0)	e(−1, 1)
e(0, −1)	e(0, 0)	e(0, 1)
e(1, −1)	e(1, 0)	e(1, 1)

$e(i,j) = 0, if\ (i,j)$ is not in the defined grid

These patterns are represented as filters, which are vectors (1-D) or matrices (2-D). For fast performance, usually the dimension of these filters are 1×3 (1-D) or 3×3 (2-D). From the point of view of functions, filters are discrete operators of directional derivatives. Instead of finding the local maxima of the gradient, we set a threshold and consider those points with gradient above the threshold as edge points. Given the source image f(x, y), the edge image E(x, y) is given by

$$E = \sqrt{(f * s)^2 + (f * t)^2}$$

Where *s* and *t* are two filters of different directions

Roberts Edge Detector:

$$s = \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & -1 \\ \hline\end{array} \qquad t = \begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline -1 & 0 & 0 \\ \hline\end{array}$$

Figure 1: Edge patterns of Roberts Edge Detector

Figure 2: Edge patterns for Roberts edge detector: (a) s; (b) t

Here the edge gradient can be obtained by forming running differences of diagonal pairs of pixels. For Roberts operator the edge gradient is shown in figure 1 and 2.

Prewitt and Sobel Edge Detector:

Consider the numbering convention for 3x3 edge detection operators

A_0	A_1	A_2
A_7	F(j, k)	A_3
A_6	A_5	A_4

$$s = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \qquad t = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

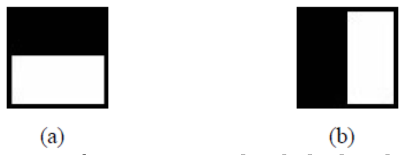

Figure 3: Edge patterns for Prewitts and Sobel edge detector: (a) s; (b) t

These filters have longer support. They differentiate in one direction and average in the other direction. So the edge detector is less vulnerable to noise. However, the position of the edges might be altered due to the average operation.

Laplacian of Gaussian:

The Laplacian is a 2-D isotropic measure of the 2nd spatial derivative of an image. The Laplacian of an image highlights regions of rapid intensity change and is therefore often used for edge detection. The Laplacian is often applied to an image that has first been smoothed with something approximating a Gaussian Smoothing filter in order to reduce its sensitivity to noise. The operator normally takes a single gray level image as input and produces another gray level image as output.

The Laplacian L(x, y) of an image with pixel intensity values I(x, y) is given by:

$$L(j,k) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

Since the input image is represented as a set of discrete pixels, we have to find a discrete convolution kernel that can approximate the second derivatives in the definition of the Laplacian.

Three commonly used small kernels are given by

1	1	1		-1	2	-1		0	1	0
1	-8	1		2	-4	2		1	-4	1
1	1	1		-1	2	-1		0	1	0

Because these kernels are approximating a second derivative measurement on the image, they are very sensitive to noise. To counter this, the image is often Gaussian Smoothed before applying the Laplacian filter. This pre-processing step reduces the high frequency noise components prior to the differentiation step. In fact, since the convolution operation is associative, we can convolve the Gaussian smoothing filter with the Laplacian filter first of all, and then convolve this hybrid filter with the image to achieve the required result. Doing things this way has two advantages: Since both the Gaussian and the Laplacian kernels are usually much smaller than the image, this method usually requires far fewer arithmetic operations.

Canny Edge Detector:

The Canny edge detection algorithm is known to many as the optimal edge detector. Canny's intentions were to enhance the many edge detectors already out at the time he started his work. He was very successful in achieving his goal and his ideas and methods can be found in his paper, "A Computational Approach to Edge Detection". In his paper, he followed a list of criteria to improve current methods of edge detection.

- The first and most obvious is low error rate. It is important that edges occurring in images should not be missed and that there be no responses to non-edges.
- The second criterion is that the edge points be well localized. In other words, the distance between the edge pixels as found by the detector and the actual edge is to be at a minimum.
- A third criterion is to have only one response to a single edge. This was implemented because the first two were not substantial enough to completely eliminate the possibility of multiple responses to an edge.

Based on these criteria, the canny edge detector first smoothen the image to eliminate and noise. It then finds the image gradient to highlight regions with high spatial derivatives. The algorithm then tracks along these regions and suppresses any pixel that is not at the maximum (non maximum suppression). The gradient array is now further reduced by hysteresis.

Hysteresis is used to track along the remaining pixels that have not been suppressed. Hysteresis uses two thresholds and if the magnitude is below the first threshold, it is set to zero (made a non-edge). If the magnitude is above the high threshold, it is made an edge. And if the magnitude is between the 2 thresholds, then it is set to zero unless there is a path from this pixel to a pixel with a gradient above T2.

Step 1:- In order to implement the canny edge detector algorithm, a series of steps must be followed. The first step is to filter out any noise in the original image before trying to locate and detect any edges. And because the Gaussian filter can be computed using a simple mask, it is used exclusively in the Canny algorithm. Once a suitable mask has been calculated, the Gaussian smoothing can be performed using standard convolution methods. A convolution mask is usually much smaller than the actual image. As a result, the mask is slid over the image, manipulating a square of pixels at a time. The larger the width of the Gaussian mask, the lower is the detector's sensitivity to noise. The localization error in the detected edges also increases slightly as the Gaussian width is increased.

Step 2:- After smoothing the image and eliminating the noise, the next step is to find the edge strength by taking the gradient of the image. The Sobel operator performs a 2-D spatial gradient measurement on an image. Then, the approximate absolute gradient magnitude (edge strength) at each point can be found. The Sobel operator [3] uses a pair of 3x3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y-direction (rows).

They are given by

G_R				G_C		
1	2	1		1	0	-1
0	0	0		2	0	-2
-1	-2	-1		1	0	-1

The magnitude, or edge strength, of the gradient is then approximated using the formula:

$$G(j,k) = \sqrt{G_R(j,k)^2 + G_C(j,k)^2}$$

Step 3:- The direction of the edge is computed using the gradient in the x and y directions. However, an error will be generated when sum X is equal to zero. So in the code there has to be a restriction set whenever this takes place. Whenever the gradient in the x direction is equal to zero, the edge direction has to be equal to 90 degrees or 0 degrees, depending on what the value of the gradient in the y-direction is equal to. If GY has a value of zero, the edge direction will equal 0 degrees. Otherwise the edge direction will equal 90 degrees. The formula for finding the edge direction is just:

$$\theta(j,k) = arcran\frac{G_C(j,k)}{G_R(j,k)}$$

Step 4:- Once the edge direction is known, the next step is to relate the edge direction to a direction that can be traced in an image.

Step 5:- After the edge directions are known, non-maximum suppression now has to be applied. Non-maximum suppression is used to trace along the edge in the edge direction and suppress any pixel value (sets it equal to 0) that is not considered to be an edge. This will give a thin line in the output image.

Step 6:- Finally, hysteresis is used as a means of eliminating streaking. Streaking is the breaking up of an edge contour caused by the operator output fluctuating above and below the threshold. If a single threshold, T1 is applied to an image, and an edge has an average strength equal to T1, then due to noise, there will be instances where the edge dips below the threshold. Equally it will also extend above the threshold making an edge look like a dashed line. To avoid this, hysteresis uses 2 thresholds, a high and a low. Any pixel in the image that has a value greater than T1 is presumed to be an edge pixel, and is marked as such immediately. Then, any pixels that are connected to this edge pixel and that have a value greater than T2 are also selected as edge pixels. If you think of following an edge, you need a gradient of T2 to start but you don't stop till you hit a gradient below T1.

DIGITAL SIGNAL PROCESSOR: The TMS320C64x™ DSPs (including the TMS320DM642 device) are the highest-performance fixed-point DSP generation in the TMS320C6000™ DSP platform. The TMS320DM642 (DM642) device is based on the second-generation high-performance, advanced VelociTI™ very-long-instruction-word (VLIW) architecture (VelociTI.2™) developed by Texas Instruments (TI), making these DSPs an excellent choice for digital media applications. The C64x™ is a code-compatible member of the C6000™ DSP platform.

III. IMPLEMENTATION DETAILS

The experiments are conducted at NMIT Research center for different sets of input video of various file formats. The different edge detection algorithms are considered for experimentation. The digital signal processor multimedia developer kit DM 642 operating at 600 MHZ is considered for the experimentation. All the source files are written in C programming language. The DM642 DSP is capable of executing eight instructions in parallel at the clock speed of 600 MHz If we are able to utilize these execution units then we can benefit from its processing power and obtain a high performance solution. Otherwise, the performance may not be satisfactory. The un-optimized codes for different interpolation techniques are considered for validation.

IV. EXPERIMENTAL RESULTS AND DISCUSSIONS

The test set for this evaluation experiment is performed on DSP Board DM 642 with Code Composer Studio. The experiment is conducted on several images of different formats, dimensions and different types. The different biomedical images such as X ray, EEG, MRI and CT are considered for the experimentation and the different segmentation algorithms are compared. The sample of the result is displayed by considering the images such as cameraman image of TIFF format and size is 256 x 256, X-Ray of lungs of BMP format of size 673 x 743 and color image of fruits in JPEG format of size 297 x 233 shown in figure 4.

	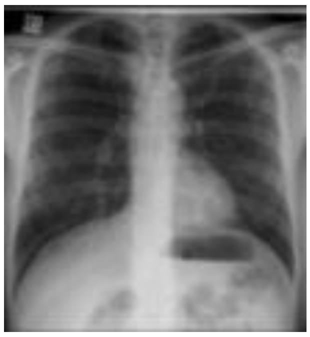	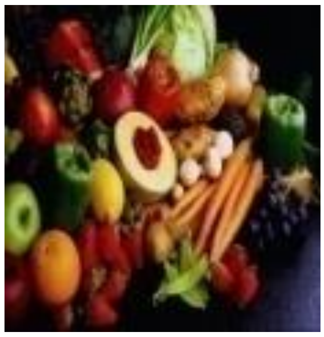
CAMERAMAN 256 x 256	LUNGS (X RAY) 673 x 743	FRUITS IMAGE 297 x 233

Figure 4: Input Images with descriptions

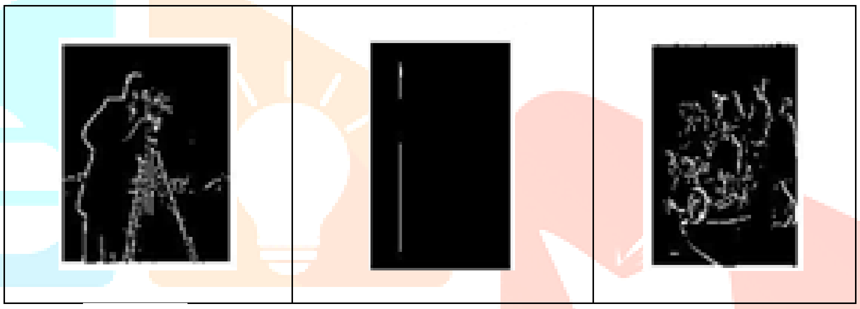

Figure 5: Image segmentation using ROBERT CROSS operators

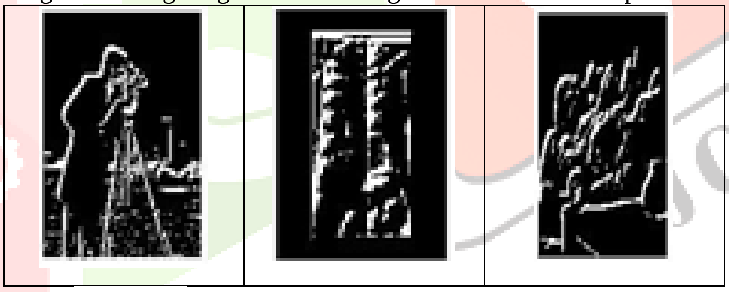

Figure 6: Image segmentation using SOBEL operators

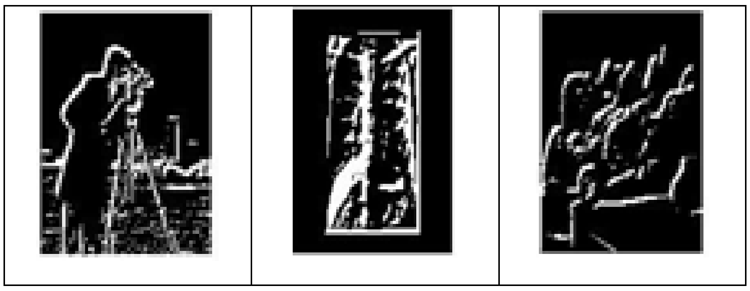

Figure 7: Image segmentation using PREWITT operators

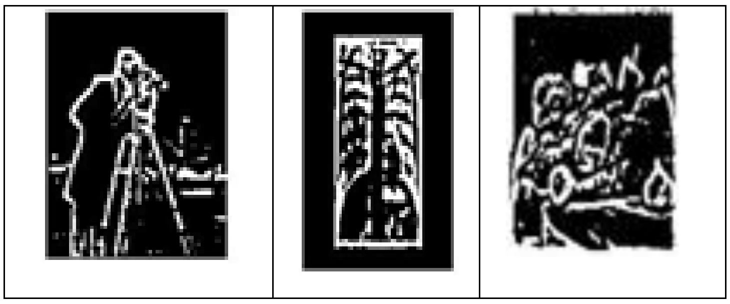

Figure 8: Image segmentation using LOG operators

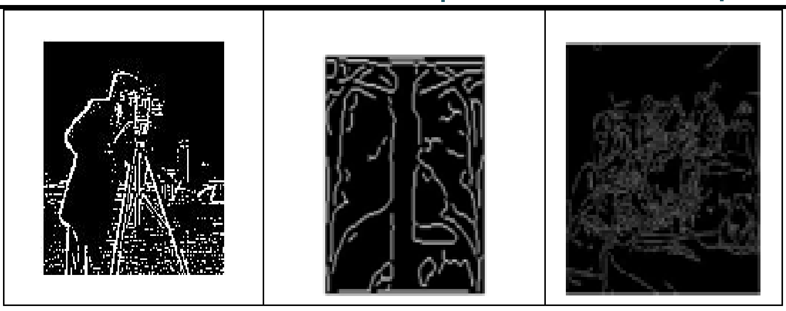
Figure 9: Image segmentation using Canny operators

Table 1: Computational requirement of edge detection operators

Edge detection operator	Sobel	Prewitt	Roberts	LOG	Canny
Memory	5.93	5.93	5.93	10.6	20
Computation time	0.16	0.16	0.16	0.54	0.3

From the figure 5 to 9 and Table 1, it is noted that for classical operators such as Sobel, Prewitt and Roberts cross, shifting operation has consumed maximum memory because operation for gradients of both horizontal and vertical directions are taken in same function. For Laplacian of Gaussian (LOG) operator, convolution operation consumes maximum memory because the convolution operation is done twice, one for Laplacian filter and Gaussian filter, and another for LOG filter and the image. For canny operator, convolution and thresholding consumes more memory, because this operator will do hysteresis thresholding

V. SUMMARY AND CONCLUSIONS

The choice of the edge detection technique in most cases depends on the application and image in question rather than a generalized method. A real-time implementation of edge detection techniques on gray and color images image signals using different algorithms are presented. The algorithms are implemented using TI TMS320DM642 imaging developer kit. The results show that the execution time is low while the edge results are accurate.

REFERENCES

1. Gunasheela K S, H S Prasantha, “Satellite image compressiondetailed survey of the algorithms”, Proceedings of ICCR in LNNS Springer, 2017, vol. 14, pp. 187-198.
2. Bellon, O.R.P. Silva, L.” New improvements to range image segmentation by edge detection” Signal Processing Letters, IEEE Volume 9, Issue 2, Feb. 2002 Page(s):43 -45
3. J. F. Haddon, “Generalized threshold selection for edge detection Pattern Recognition.”, vol. 21, pp. 195-203, 1988.
4. Ikhlas Abdel Qader, Ph.D., P.E , Marie Maddix, “Real-Time Edge Detection Using TMS320C6711 DSP”,Electro/information technology conference, 2004,EIT 2004,IEEE, pp 306 – 309.
5. H. S. Prasantha, H. L. Shashidhara, and K. N. Balasubramanya Murthy. Image compression using SVD. In Proceedings of the International Conference on Computational Intelligence and Multimedia Applications, pages 143–145. IEEE Computer Society, 2007.
6. M. J. Raghavendra, H. S. Prasantha and S. Sandya, "Image Compression Using Hybrid Combinations of DCT SVD and RLE", International Journal of Computer Techniques, vol. 2, no. 5, 2015.
7. Gunasheela K S, H S Prasantha, “Compressive sensing for image compression: survey of algorithms”, Proceedings of Emerging Research in Computing, Information, Communication and Applications, ERCICA, Springer publication, Bengaluru, 2018
8. K N Shruthi, B M Shashank, Y. SaiKrishna Saketh, H.S Prasantha and S. Sandya, "Comparison Analysis Of A Biomedical Image For Compression Using Various Transform Coding Techniques", IEEE, pp. 297-303, 2016
9. Canny, J., "A Computational Approach to Edge Detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 9, Number 6, 1986, Pages 679-698.
10. Heath, M., Sarkar, S., Sanocki, T., and Bowyer, K., "Comparison of Edge Detectors: A Methodology and Initial Study," Journal of Computer Vision and Image Understanding, Volume 69, Number 1, 1998.
11. M.J. Raghavendra, H. S. Prasantha and S. Sandya, "DCT SVD Based Hybrid Transform Coding for Image Compression", International Journal of Recent and Innovative Trends in computing and communication, 2015.
12. H. S Prashanth, HL Shashidhara and Murthy KNB, "Comparative analysis of different interpolation schemes in image processing", International Conference on Advanced Communication Systems, January 10–12, 2007.
13. Dr. H S Prasantha, “NOVEL APPROACH FOR IMAGE COMPRESSION USING MODIFIED SVD”, International Journal of Creative Research Thoughts (IJCRT), Volume 8, Issue 8, Page 2234-2243, Aug 2020
14. Dr. H S Prasantha, “IMPLEMENTATION OF IMAGE COMPRESSION USING FAST COMPUTATION OF SVD ON DM642”, International Journal of Creative Research Thoughts (IJCRT), Volume 8, Issue 8, Page 2364-2368, Aug 2020
15. Prasantha, H, H Shashidhara, K N B Murthy, and M Venkatesh. "Performance Evaluation of H.264 Decoder on Different Processors." International Journal on Computer Science & Engineering. 1.5 (2010): 1768. Web. 7 Apr. 2013.