IJCSN - International Journal of Computer Science and Network, Volume 3, Issue 6, December 2014
ISSN (Online): 2277-5420 **www.IJCSN.org**
Impact Factor: 0.274

Real Time Edge Detection Using DSP Processor

[1] Shashank Navale, [2] Pooja P. Gundewar

[1] Department of Electronics and Telecommunication
MIT College of Engineering, Pune, India

[2] Department of Electronics and Telecommunication
MIT College of Engineering, Pune, India

Abstract - Edge detection from images is one of the most important concerns in digital image and video processing. Edge detection is very useful in computer vision for extracting meaningful information from images. Real time edge detection is implemented on the platform that consists of the TMS320DM6437 DSP, camera, TV tuner, Display and canny edge detection algorithm. The main goal of our technique is to obtain thin edges. The edge detection is useful in image segmentation, object identification and boundary detection. Edge detection is useful for extracting information about the image like location of object in the image their shape and size. Image edge detection reduces the amount of data and filters out useless information and preserves important structural properties in an image. The edge image contains information about the original image. The system is working on real time and it does not require any additional sensor input except an image.

Keywords - *Real time edge detection, canny edge detection, TV tuner, DSP Processor.*

1. Introduction

Image processing has been widely used in a range of industrial, commercial, civilian and military applications. Edges are the significant local changes in intensity of an image. An edge is defined as pixel intensity discontinuity or irregularities. The edge was a boundary of different homogenous regions. Edges carry critical information about the scene. Most of the information in a scene is contained in the edge information. Edge detection is an ongoing research topic in image analysis and computer vision. Edge detection refers to the process of identifying and locating sharp discontinuities in an image. Edge detection is always the first step in many image processing algorithms, because it significantly reduces the amount of data and filters out useless information. Most of the information in a scene is contain in the edge information. The purpose of edge detection in general is to reduce the amount of data in an image and preserves the structural properties to be used for further image processing. The edge is the set of those pixels. The gray, texture, or the color of those pixels is different with the adjacent pixels in the image. It is a necessary step for information and feature extraction. Edge detection is the important basis in image analysis and image recognition. The real time image edge detection is one of the main objectives in image recognition. The goal of this project is to implement an image processing algorithm applicable to edge detection system in DSP, with a focus on achieving overall high performance and short development time. The aim of edge detection is to develop an algorithm that is optimal with regards to the following criteria:

1. *Detection:* It should be detect real edge point while the falsely detecting non edge points should be less.
2. *Localization:* The detected edges should be as close as to the real edges.
3. *Number of responses:* One real edge should not give more than one detected edge.

There are many algorithms for edge detection such as Sobel, Roberts Cross, Prewitt, and Canny but Canny is superior over many of the available algorithm and thus is chosen for real time implementation. They can show where shadow fall in an image or any other distinct change in the intensity of an image. The quality of edge detection is highly dependent on the condition of light, the objects of similar intensities, density of edges in the image, and noise. While each of these problems can be handled by adjusting certain values in the edge detector and changing the threshold value for what is considered an edge. There are two ways for implementation of edge detection, one using intensity and the other using color information. Canny algorithm based real time edge detection method is use to improve the processing speed. In this project we use a camera which takes multiple images under different lighting conditions. The method is based on DSP processor and the canny edge detection. The system can be implemented in the real time system for the mist, smoke,

dust, and rainy conditions. In presently global market for video processing systems requires high performance digital signal processing as well as low device costs appropriate for a volume application. DSP device provide a platform with which to meet these two contrasting requirements. The paper proposed a new method for implementing the real time edge detection rapidly based on the platform of TMS320DM6437 DSP processor. The application of this system is in industrial assembly lines and surveillance systems for finding the intruder.

The program of canny edge detection is in the CCS (Code Compose Studio) language programming environment. In section II related work is discuss. In section III Hardware implementation are described. In section IV Edge detection is describe. In section V Canny edge detector is described. In section VI Experimental result is present. The conclusion is presented in section VII.

2. Related Work

In recent year many algorithms has been developed for edge detection techniques. Obili Ramesh et al. have proposed architecture for Edge Detection using Sobel Filter & Canny Filter for image processing using Xilinx system generator. The goal of this project is to implement an image processing algorithm applicable to Edge Detection system in a Xilinx FPGA using system generator for DSP. In this paper author define the canny edge detection method which gives sharp edge image compared to the Sobel edge detection method. High performance, low cost and short development time is achieved using this method. In this paper camera is not use so we cannot implement it for real time [1].

Tomasz Marciniak et al. have proposed a concept for fast prototyping of real-time hardware/software video processing systems for urban surveillance monitoring equipment. The evaluation module with the TMS320DM6437 signal processor linked with the Code Composer Studio through Matlab/Simulink has been used. The processed video signals have been acquired using BOSCH NBC-255-P network camera with the CMOS ¼" sensor. The author analyzed efficiency of implementation of the algorithms using two examples: detection of painting theft and signaling of crossing a pedestrian pass at the red light. In this paper motion detection, edge detection, color segmentation, people tracking abandoned object detection models are used. The real-time implementations of both models have been done using the TMS320DM6437 EVM. Concluding, the use of the Matlab/Simulink environment provides a very convenient opportunity for creating, modifying, and testing software based on the C6000 processor family. The module architecture allows the optimal implementation of the algorithms such as tracking, classification, and detection of objects in real time. Results of multiple tests allow to conclude that the use of the Matlab/Simulink environment and the CCS require the continuous code optimization to implement the system with the highest possible processing rate. Currently, the image processing within our system runs at 4 fps, which is adequate to the speed of the image registration in contemporary surveillance systems, but it may be too slow in advanced systems working with larger values of fps [2].

Jinqing Liu et al. have proposed a system for human face detection This system firstly summarizes the basic methods of edge detection and the basis of Sobel operator and Canny operator, then, from a practical perspective, puts forward an novel method that is based on the integration of improved Sobel operator and Canny operator, moreover its results are thinned lastly via the method of Improved OPTA. All of that above is implemented on TMS320 DM642EVM. The results testify that this approach is not only good at eliminating noise, but also can detect the edge quickly and completely, so it has well practical significance. All of that is a foregoing preparation for the achievement of the system of human face detection in application. The integrity and real-time of detection have made some improvement in this paper, however, in real-time image processing system, how to achieve the ingenious combination between traditional measures and morphology etc, how to detect edges of human face exactly and real-time in the realistic environment containing such as noise, uneven illumination and so on, which needs continuous exploration in the application field [3].

Ikhlas Abdel Qader et al. have proposed a method for Real-Time Edge Detection Using TMS320C6711 DSP. The goal of this project is a real-time implementation of edge detection in grayscale image signals using the canny algorithm and DSP. The results show that the execution time is low while the edge results are accurate and thus presenting a suitable algorithm for on-line vision systems applications such as industrial assembly lines, surveillance systems (intruder detection), and on-line hand written document reading. Each pixel requires that the square root be taken for the magnitude and the inverse tangent to be taken for the direction. These floating-point operations are costly on the processor. To reduce computation time, the thresholds T I and T2 can be altered to reflect the squared value of the magnitude without a great loss in results quality. The algorithm can be used for more complex scenes such as applications in surveillance and detection of intruders [4]. Parvinder Singh Sandhu et al. have proposed technique to obtain thin edges, so that the result is more suitable for further application such as boundary detection, object identification. The technique of edge detection

tested four edge detectors that use different methods for detecting edges and compared their results under a variety of situations to determine which detector was preferable under different sets of conditions. For one of the edge detectors the author considered two different ways of implementation, the one using intensity only and the other using color information. For the Marr Hildreth edge detector, it is possible to set the slope threshold, the sigma and size of the Gaussian. For the Canny edge detector it is possible to set the high threshold and low threshold, the sigma and size of the Gaussian. For the Euclidean distance and vector angle color edge detector, it is possible to set the slope and offset, and to set the final threshold. For the Multi-Flash edge detector, it is possible to set the threshold of the negative edge step. In the case of the canny color edge detector, it usually finds more edges than the grayscale version. The canny edge detector becomes fairly confused at corners due to the Gaussian smoothing of the image. Also, since the direction of the edge changes instantly, corner pixels looks in the wrong directions for its neighbors [5].

From the above edge detection technique we can conclude that the canny edge detection gives sharp edges and better performance than the other edge detection algorithm. The propose system uses DSP processor which gives better performance and increases the speed. The DSP reduces the transmission delay for large sustained bandwidth. The comparison of Sobel edge detection and canny edge detection technique is shown in table 1.

Table 1 : Comparison of Edge Detection Techniques

Parameters	Sobel	Canny
Computation time	Time efficient	Time consuming
Complexity	Simple	Complex
Signal to noise ratio	Low	High
Quality of an image (Texture based image)	Less efficient	More efficient
No of object in image	Suitable for simple images	Suitable for simple as well as complex images
Area of application	Massive data transfer	For X-ray diagnosis in medical field

3. Hardware Setup

The real time edge detection system is based on the platform of DSP processor. The three ends of the DSP are linked to the computer, TV tuner and camera. The one port of DSP is link to a computer. The input port of the DSP processor is linked to the camera and output port is connected to the TV tuner. The 'C6437 is a high-performance, 32 bit floating point DSP processor developed by Texas Instruments. The speed of this processor makes it well suited for image processing applications. The C64x+ DSP core processor has 64 general-purpose registers of 32-bit word length and eight highly independent functional unit two multipliers for a 32-bit result and six arithmetic logic units. The eight functional units include instructions to accelerate the performance in video and image applications. The DM6437 also has application specific hardware logic, on-chip memory, and additional on-chip peripherals similar to the other C6000 DSP platform devices. The DM6437 core uses a two-level cache-based architecture. The Level 1 program memory/cache (L1P) consists of a 256K-bit memory space that can be configured as mapped memory or direct mapped cache, and the Level 1 data (L1D) consists of a 640K-bit memory space. The Level 2 memory/cache (L2) consists of a 1M-bit memory space that is shared between program and data space. L2 memory can be configured as mapped memory, cache, or combinations of the two [12]. The hardware set up is shown in Figure 1 and experimental set up is shown in Figure 2.

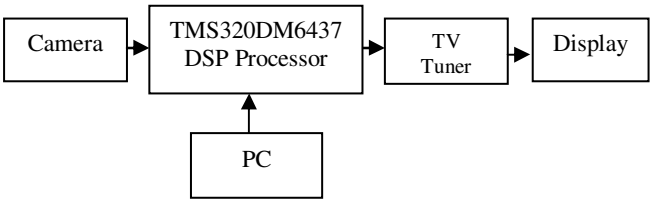

Figure 1: Hardware used in this system

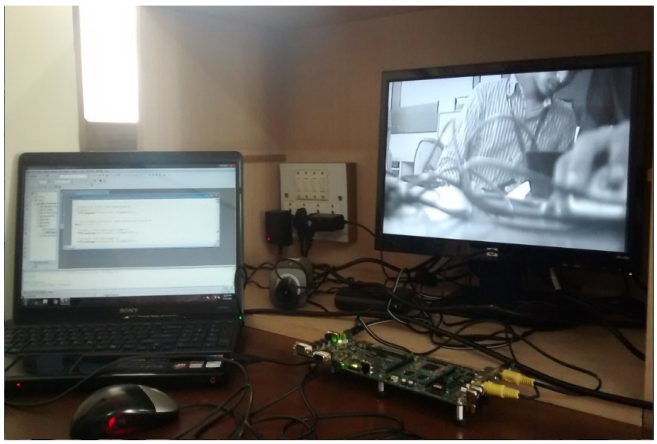

Figure 2: Experimental set up

4. Edge Detection

The points at which image brightness changes sharply are typically organized into a set of curved line segments termed edges. Edge detection is a set of mathematical methods which identifying points in a digital image at which the image brightness changes sharply or more formally has discontinuities. Edge detection is useful in image processing, machine vision, computer vision, particularly in the areas of feature detection and feature extraction. The desired edges are the boundaries between adjacent intensity. The block diagram of edge detection is shown in figure 3.

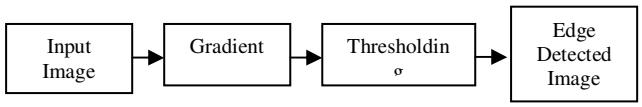

Figure 3: Block diagram of edge detection

5. Canny Edge Detection

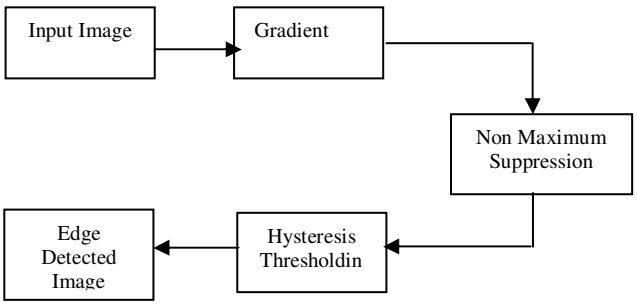

Figure 4: Block diagram of canny edge detection algorithm

The Canny edge detector is widely used edge detection algorithm in the industry. The Canny edge detector is an edge detection operator that uses a multi stage algorithm to detect a wide range of edges in images. The aim of Canny is to discover the optimal edge detection algorithm. In this situation, an "optimal" edge detector means:

1. *Good Detection* - The algorithm should mark as many real edges in the image as possible.
2. *Good Localization* - Edges marked should be as close as possible to the edge in the real image.
3. *Minimal Response* - A given edge in the image should only be marked once, and image noise should not create false edges.

To satisfy these requirements canny used the calculus of variations a technique which finds the function which optimizes a given functional. The optimal function in Canny's detector is described by the sum of four exponential terms, but it can be approximated by the first derivative of a Gaussian. The canny edge detector is one of the most commonly used image processing tools, detecting edges in a very robust manner. The block diagram of edge detection is shown in figure 4.

The algorithm runs in 4 separate steps:

1. Smoothing: Blurring of the image to remove noise.
2. Finding gradients: The edges should be marked where the gradients of the image have large magnitudes.
3. Non maximum suppression: Only local maxima should be marked as edges.
4. Hysteresis thresholding: Potential edges are determined by thresholding. Final edges are determined by suppressing all edges that are not connected to a strong edge.

Each step is described in the following subsections.

Step 1: Smoothing

Generally noise reduction is essential factor in image processing. An image is always affected by noise in its capture, acquisition and processing. To prevent mistaken of edges, noise must be reduced. Therefore the image is first smoothed by applying a Gaussian filter. Smooth the image with a two dimensional Gaussian. The input image is smoothed to eliminate noise by improved Gaussian filter. Smoothing the image is done using a Gaussian mask. The Gaussian mask is created using either equation one or two.

$$G(x) = \frac{1}{\sqrt{2\pi\,\sigma}} e^{-\frac{x^2}{2\sigma^2}} \quad [1]$$

$$G(x, y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad [2]$$

The first creates a weighted 1-D mask based on the standard deviation and neighboring pixels in the horizontal direction. The second formula creates a 2-D mask similar to the first but is it also dependent upon the pixels in the vertical direction. Usually noise reduction implies some sort of blurring operation. Gaussian filter is use to do this.

Step 2: Compute Gradient Magnitude and Angle

The Canny algorithm basically finds edges where the grayscale intensity of the image changes the most. These areas are found by determining gradients of the image. First step is to approximate the gradient in the x- and y-direction respectively by applying the kernels the first order derivative is applied to the entire image in both horizontal and vertical directions. For any given pixel located at (x, y), can be calculated using following equations.

$$Gx = Valx+1 - Valx-1 \quad [3]$$

$$Gy = Valy+1 - Valy-1 \quad [4]$$

The magnitude and direction of each pixel at (x, y) are computed using

$$|G| = \sqrt{Gx^2 + Gy^2} \quad [5]$$

$$|G| = |Gx| + |Gy| \quad [6]$$

Where, Gx and Gy are the gradients in the x- and y-directions respectively. An image of the gradient magnitudes often indicates the edges quite clearly. However, the edges are typically broad and thus do not indicate exactly where the edges are

$$\theta = \arctan\left(\frac{Gy}{Gx}\right) \quad [7]$$

This shows changes in intensity, which indicates the presence of edges. This actually gives two results, the gradient in the x direction and the gradient in the y direction. An edge in an image may point in a variety of directions, so the canny algorithm uses four filters to detect horizontal, vertical and diagonal edges in the blurred image. The edge direction angle is rounded to one of four angles representing vertical, horizontal and the two diagonals that are 0, 45, 90 and 135 degrees.

Step 3: Non Maximum Suppression

The purpose of this step is to convert the blurred edges in the image of the gradient magnitudes to sharp edges. Basically this is done by preserving all local maxima in the gradient image, and deleting everything else. The non-maximal suppression step keeps only those pixels on an edge with the highest gradient magnitude. Edges will occur at points where the gradient is at a maximum. Therefore, all points not at a maximum should be suppressed. In order to do this, the magnitude and direction of the gradient is computed at each pixel.

Step 4: Hysteresis Thresholding

The edge pixels remaining after the non maximum suppression step are marked with their strength pixel by pixel. Many of these will probably be true edges in the image, but some maybe caused by noise or color variations for instance due to rough surfaces. The simplest way to discern between these would be to use a threshold, so that only edges stronger that a certain value would be preserved. A simple threshold may actually remove valid parts of a connected edge, leaving a disconnected final edge image. This happens in regions where the edge's gradient magnitude fluctuates between just above and just below the threshold. Hysteresis is one way of solving this problem. Instead of choosing a single threshold, two thresholds thigh and tlow are used. Pixels with a gradient magnitude D < tlow are discarded immediately. However, pixels with tlow D < thigh are only kept if they form a continuous edge line with pixels with high gradient magnitude.

- If pixel (x, y) has gradient magnitude less than tlow discard the edge
- If pixel (x, y) has gradient magnitude greater than thigh it is set as edge.
- If pixel (x, y) has gradient magnitude between tlow and thigh and a pixel has a value above the low threshold and is the neighbor of an edge pixel, it is set as an edge pixel.
- If pixel (x, y) has gradient magnitude between tlow and thigh and a pixel has a value above the low threshold but is not the neighbor of an edge pixel, it is not set as an edge pixel[4][8].

6. Experimental Result

The preceding algorithm is written in Code Composer Studio for images on the TMS320DM6437 DSP. Several implementation issues need to be addressed. First, read the each pixel of image in horizontal and vertical direction. Second, convert the color image into gray scale. Third, smooth the image. Fourth edge pixels are assigned the exact same location they are detected at. Finally, display the image. In this paper firstly canny edge detection is implemented using C language. The results in C language of the Figure 5 are showing below.

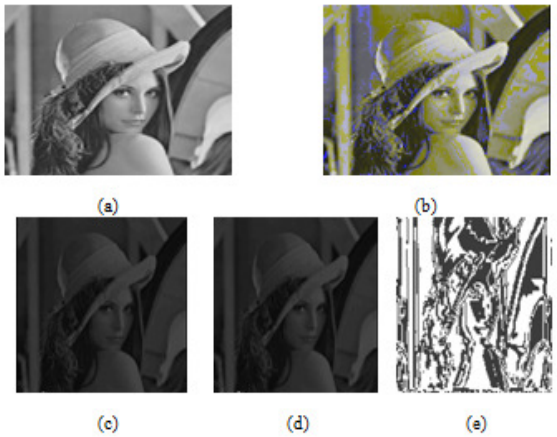

Figure 5: (a) Original Image, (b) Observe Image, (c) Gray scale Image, (d)Smooth Image, (e) Threshold Image

The results in DSP processor of the Figure 6 are showing below.

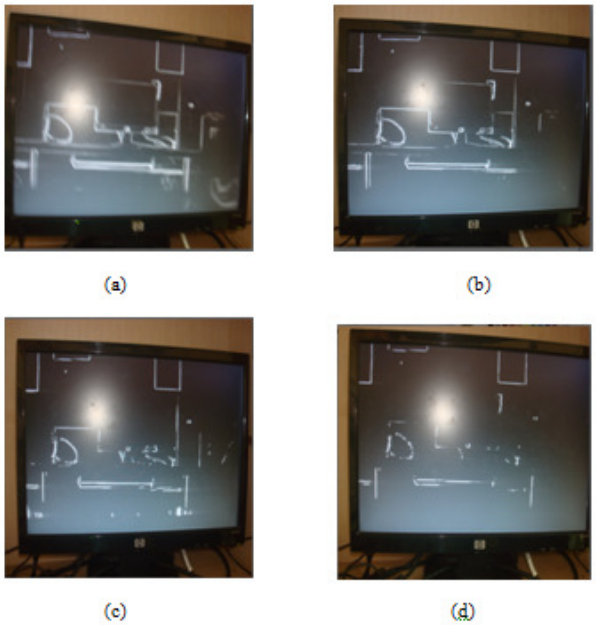

Figure 6: (a) Threshold 60, (b) Threshold 90, (c) Threshold 120, (d) Threshold 150

7. Conclusion

The method of the real-time edge detection is good. The nearer objects real time edge detection is satisfactory. Real time edge detection is a preprocess for many critical applications such as assembly line inspection and surveillance. The system is run in real time. The observe image contain some noise. The canny edge detection method gives sharp edge image and the imaging developer board which is used for real time implementation of image processing algorithms is provided higher resolution and maximum speed.

References

[1] Obili Ramesh, P. V. Krishna Mohan Gupta, B. Sreenivasu,“ A Real Time Hardware and Software Co-Simulation of Edge Detection for Image Processing System”, International journal of engineering Research and Technology Vol. 2, Issue 8, Year: August 2013.

[2] Tomasz Marciniak, Agata Chmielewska, Rados_aw Weychan, Adam D_browski, “Fast Prototyping of Automatic Real-time Event Detection Facilities for Video Monitoring Using DSP Module”, 20th International Conference on Mixed Design of Integrated Circuits and Systems, Year: June 20-22, 2013.

[3] Jinqing Liu, Yusheng Huang, “Study of Human Face Image Edge Detection based on DM642”, IEEE International Conference on Computer Science and Information Theory, volume 8, Year: 2010.

[4] Ikhlas Abdel Qader, Marie Maddix, “Real-Time Edge Detection Using TMS320C6711 DSP”, Electro/Information Technology Conference, IEEE, Year: 2004.

[5] Parvinder Singh Sandhu, MamtaJuneja, EktaWalia, “Comparative Analysis of Edge Detection Techniques for extracting Refined Boundaries”, International Conference on Machine Learning and Computing Volume(3), Year:2009.

[6] JIANG Xingfang, BI Tianyu, TAO Chunkan, “A Method of Real-time Edge detection”, International Conference on Electric Information and Control Engineering, Year: 2011.

[7] Zhiwei TANG, Dongqin SHEN,“ Canny Edge Detection Codec using VLib on davinci series DSP”, International Conference on Computer Science and Service System, Year: 2012.

[8] Raman Maini, Himanshu Aggarwal, “Study and Comparison of Various Image Edge Detection Techniques**”,**International Journal of Image Processing (IJIP), Volume (3), Issue (1).

[9] Huili Zhao, Guofeng Qin, Xingjian Wang, “Improvement of Canny Algorithm Based on Pavement Edge Detection” 3rd International Congress on Image and Signal Processing, Year: 2010.

[10] Kuang hang, “Real-Time Image Acquisition and Processing System Design based on DSP”, 2nd International Conference on Computer and Automation Engineering, Volume 4, Year: 2010.

[11] Clemen Arth, Florian Limberger, Horst Bisch of “Real-Time License Plate Recognition on an Embedded DSP-Platform”, IEEE Conference on Computer Vision and Pattern Recognition, Year: 2007.

[12] Texas Instruments. “TMS320DM6437 Digital Media Processor”, Year: JUNE 2008.

[13] Dwayne Phillips, “Image processing in C”, Year: April 2000.