Used of Various Edge Detection Operators for Feature Extraction in Video Scene

Ramesh R. Manza
Department of CS and IT,
Dr. B. A.M. University,
Aurangabad (MS), India.
manzaramesh@gmail.com

Bharatratna P. Gaikwad
Department of CS,
Vivekanand College of Science,
Aurangabad (MS), India
bharat.gaikwad08@gmail.com

Ganesh R.Manza
Department of CS and IT,
Dr. B. A.M. University,
Aurangabad (MS), India.
ganesh.manza@gmail.com

Abstract-This paper presents the implementation of an adaptive edge-detection using Simulink. Simulink is a simulation modeling and design tool and GUI based diagram environment. The Simulink based customizable framework is designed for rapid simulation, implementation, and verification of video and image processing algorithms and systems. An edge is the boundary between an object and the background, and indicates the boundary between overlapping objects.The Edge detection process detects outlines of an object, scene text and boundaries between objects and the background in the video image. Edge detection is in the forefront of video processing for object detection, it is crucial to have a good understanding of edge detection methods. Edges characterize boundaries and edge detection is one of the most difficult tasks in image processing hence it is a problem of fundamental importance in image processing.In this work the comparative analysis of various video image edge detection methods is like Sobel, Prewitt, and Canny is presented.

Keywords- Edge Detection**,** Sobel, Prewitt, Canny, Simulink.

I. INTRODUCTION:

Simulink is a platform for multidomain simulation and model-based design of dynamic system. Simulation is an interactive tool for modeling simulating and analyzing dynamic multi domain system. You can build a block diagram, simulate system behavior evaluate performance and refine the design .MATLAB provides you to immediate access to an extensive range of analysis and design tool. Simulink is ideal for control system design, Digital signal processing design, communication system design simulation option[2][5]. .Edge detection is a fundamental tool used in most image processing applications to obtain information from the frames before feature extraction and object segmentation. This process detects outlines of an object and boundaries between objects and the background in the image. An edge detection filter is also used to improve the appearance of blurred. The basic edge detection operator is a matrix area gradient operation that determines the level of variance between different pixels. The edge detection operator is calculated by forming a matrix centered on a pixel chosen as the center of the matrix area. If the value of this matrix area is above a given threshold, then the middle pixel is classified as an edge. Examples of gradient-based edge detectors are Sobel, Prewitt, and Canny operators.

II. EDGE DETECTION

The edge detection process serves to simplify the analysis of images by drastically reducing
the amount of data to be processed, while at the same time preserving useful structural information about object boundaries. There is certainly a great deal of diversity in the applications of edge detection. Edge detectors of some kind, particularly step edge detectors, have been an essential part of many computer vision systems. There are many edge detection operators available, each designed to be sensitive to certain types of edges. Certain criteria involved in selection of an edge detection operator include edge orientation. There are mainly exist several edge detection methods Sobel [7], Prewitt [8], Canny [9].

These methods have been proposed for detecting transitions in video images.

- Edges are significant local changes of intensity in an image.
- Edges typically occur on the boundary between two different regions in an image.
- Important features can be extracted from the edges of an image (e.g., corners, lines, curves).
- Object boundary (discontinuity in depth and/or surface color and texture)
- Surface boundary (discontinuity in surface orientation and/or surface color and texture)

The Four Steps of Edge Detection

a) Smoothing: suppress as much noise as possible, without destroying the true edges.
b) Enhancement: apply a filter to enhance the quality of the edges in the image (sharpening).
c) Detection: determine which edge pixels should be discarded as noise and which should be retained (usually, thresholding provides the criterion used for detection).
d) Localization: determine the exact location of an edge (sub-pixel resolution might be required for some applications, that is, estimate the location of an edge to better than the spacing between pixels). Edge thinning and linking are usually required in this step.

III EDGE DETECTION TECHNIQUES

1.1 Sobel Edge Detection

In this work, Sobel which is an edge detection method is considered. This method is preferred will use in this work. The Sobel edge detector uses two masks, one vertical and one horizontal. These masks are generally used 3×3 matrices. Especially, the matrices which have 3×3 dimensions are used in matlab. The Sobel method finds edges using the Sobel approximation to the derivative. It returns edges at those points where the gradient of I is maximum [9]. The operator consists of a pair of 3X3 convolution kernels as shown in fig.1. These kernels are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid [6].

Gx =

-1	0	+1
-1	0	+1
-1	0	+1

Gy =

-1	-1	-1
0	0	0
1	1	1

Fig.1 Sobel edge detection operator.

$$\text{Edge Magnitude} = \sqrt{x^2 + y^2}$$

$$\text{Edge Direction} = \tan^{-1}\left(\frac{y}{x}\right)$$

Main steps in edge detection using masks:-
Smooth the input image

1) $(\hat{f}(x, y) = f(x, y) * G(x, y))$

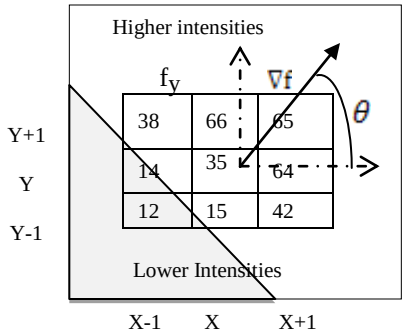
2) $\hat{f}x = \hat{f}(x, y) * Mx(x, y)$

3) $\hat{f}y = \hat{f}(x, y) * My(x, y)$

4) Magn(x, y) =| ˆf x | +| ˆf y|

5) dir(x, y) =tan-1($\hat{f}y / \hat{f}x$)

6) If $magn(x, y) > T$, then possible edge point



(An example using the Prewitt edge detector - don't divide by 2)

$f_y = ((38-12)/2 + (66-15)/2 + (65-42)/2)/3$
$= (13 + 25 + 11)/3 = 16$
$f_x = ((65-38)/2 + (64-14)/2 + (42-12/2)/3$
$= (13 + 25 + 15)/3 = 18$
$\theta = \tan^{-1}(16/18) = 0.727 \text{ rad}$
$= 42 \text{ degrees}$
$|\nabla| = (16^2 + 18^2)^{1/2} = 24$

1.2 Prewitt Edge Detection

The Prewitt method finds edges using the Prewitt approximation to the derivative. It returns edges at those points where the gradient of I is maximum. It is similar to the Sobel operator and is used for detecting Vertical and horizontal edges in video image.

Gx =

-1	+2	+1
0	0	0
-1	-2	-1

Gy =

-1	0	+1
-2	0	+2
-1	0	+1

Fig.2 Prewitt Edge detection operator.

$$\text{Edge Magnitude} = \sqrt{x^2 + y^2}$$

$$\text{Edge Direction} = \tan^{-1}\left(\frac{y}{x}\right)$$

Canny Edge Detector

The Canny method finds edges by looking for local maxima of the gradient of I. The gradient is calculated using the derivative of a Gaussian filter. The method uses two thresholds, to detect strong and weak edges, and includes the weak edges in the output only if they are connected to strong edges [12]. This method is

therefore less likely than the others to be fooled by noise, and more likely to detect true weak edges[4].

Gx =

-1	-2	-1
0	0	0
1	2	1

Gy =

-1	0	+1
-2	0	+2
-1	0	+1

Fig.3 Canny Edge detection operator.

1. Compute f_x and f_y

$$f_x = \frac{\partial}{\partial x}(f * G) = f * \frac{\partial}{\partial x} G = f * G_x$$

$$f_y = \frac{\partial}{\partial y}(f * G) = f * \frac{\partial}{\partial y} G = f * G_y$$

$G(x, y)$ *is the Gaussian function*

$G_x(x, y)$ *is the derivate of* $G(x, y)$ *with respect to* x: $G_x(x, y)$

$= (-x)/\sigma^2\ G(x, y)$

2. *Compute the gradient magnitude* $magn(i, j) = \sqrt{f_x^2 + f_y^2}$

3. *Apply non − maxima suppression.*

4. *Apply hysteresis thresholding / edge linking .*

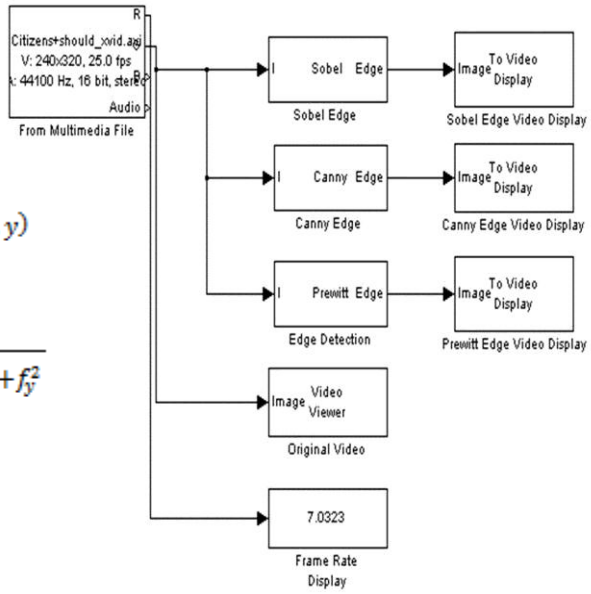
Canny's aim was to discover the optimal edge detection algorithm. In this situation, an "optimal" edge detector means:

- Good Detection – the algorithm should mark as many real edges in the image as possible.
- Good Localization – edges marked should be as close as possible to the edge in the real image.
- Minimal Response – a given edge in the image should only be marked once, and where possible, image noise should not create false edges.

IV METHODOLOGY

Implementation Analysis and Preliminary Report:

The aim of this paper work is to implement an efficient methodology to edge detection process detects outlines of an object and scene text and boundaries between objects and the background in the video image. Simulink handles stream processing implicitly by managing the flow of data through the blocks that make up a Simulink model. Simulink is an interactive graphical environment for modeling and simulating dynamic systems that uses hierarchical diagrams to represent a system model. It includes a library of general-purpose, predefined blocks to represent algorithms, Analysis & Enhancement, sources, sinks, and system hierarchy. Computer Vision System Toolbox provides a library of blocks specifically for the design of computer vision and video processing systems. In following framework various blockset are used.



a) **Sources/From Multimedia File:** On Windows reads video frames and/or audio samples from a compressed or uncompressed multimedia file. Multimedia files can contain audio, video, or audio and video data.

b) **Analysis & Enhancement/Edge Detection:** Finds the edges in an input image using Sobel, Prewitt, and Canny methods.

c) **Sinks/To Video Display:** Displays a video stream.

d) **Sinks/Frame Rate Display:** Calculate and display the frame rate of the input signal [15].

e) **Report Generator/launch report:** Clicking on this block runs the Report Generator using the "report" command. The setup file listed in the current system's "ReportName" property will be used to generate the report. If the "ReportName" property is empty, the

Report Generator will search each system parent until a non-empty "ReportName" property is found [2].

Edge detection by Prewitt, Sobel, and Canny Operator

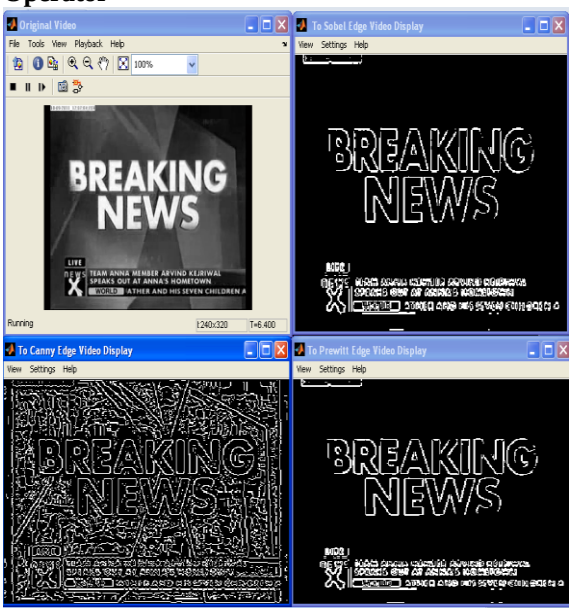


Fig.5: a) Original b) Prewitt c) Sobel d) Canny.

CONCLUSION

In this paper, the proposed operator's to Implementation of this methodology using Simulink blockset is more useful for Object Edge detection to detect outlines of an object, scene text and boundaries. Videos and images from open source can be accomplished and so it can be easily implemented and its usage is wider. Implementation of this methodology using Simulink blockset is more useful. This system provides to identify of objects and recognition scene text. The performance for edge detection in a video image is evaluated both subjectively and objectively. The subjective evaluation of edge detected video images show that proposed operator, Sobel and Prewitt and Canny operator exhibit better performances respectively.

REFERENCES

[1] The MathWorks, Simulink-Model-Based and System-Base Design. The MathWorks, September 2003. Writing S-Functions—Version 5.

[2] www.mathworks.com/res/viprocessing

[3] Hong Shan Neoh and Asher Hazanchuk,"Adaptive Edge Detection for Real-Time Video Processing using FPGAs". Altera Corporation, Copyright © 2005 Altera Corporation.

[4] Canny, J., "A Computational Approach to Edge Detection", IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, November 1986.

[5] Dr.P.Subashini,Ms.M.Krishnaveni," Implementation Of Object Tracking System Using Region Filtering Algorithm Based On Simulink Blocksets", International Journal of Engineering Science and Technology (IJEST),Vol.3 No.8 August 2011.

[6] H.K.Chethan and G.Hemantha Kumar," A Comparative Analysis of Different Edge Based Algorithms for Mobile/Camera Captured Images ", International Journal of Computer Applications (0975 - 8887) Volume 7– No.3, Sept. 2010.

[7] Sobel, I., "An Isotropic 3×3 Gradient Operator, Machine Vision for Three – Dimensional Scenes", Freeman, H.,Academic Pres, NY, 376-379, 1990.

[8] Prewitt, J., "Object Enhancemet And Extraction, Picture Processing and Psychopictorics" (B. Lipkin and A. Rosenfeld,
Ed.), NY, Academic Pres, 1970.

[9] Canny,J., "A Computational Approach to Edge Detection", IEEE Transactions on Pattern Analysis anad Machine Intelligence, 8, 679-700, 1986.

[10] J. Canny,"Finding edges and lines in image". Master's Thesis, MIT, 1983.

[11] Mamta Juneja , Parvinder Singh Sandhu," Performance Evaluation of Edge Detection Techniques for Images in Spatial Domain", International Journal of Computer Theory and Engineering, Vol. 1, No. 5, December, 2009, 1793-8201.

[12] Y. Yakimovsky, "Boundary and object detection in real world images". JACM, vol. 23, no. 4, pp. 598-619, Oct. 1976.

[13] Ziou, D. and S. Tabbone, "Edge detection techniques an overview", International Journal of Pattern Recognition Image Analysis, vol. 8: 537-559, 1998.

[14] Shervan Fekri Ershad, "Texture Classification Approach Based on Combination of Edge & Co-occurrence and Local Binary Pattern"

[15] Stauffer C.; "Estimating tracking sources and sinks", Proc of 2nd IEEE Workshop on Event Mining (in conjunction with CVPR'2003), 4, Madison, Wisconsin (June 2003).

[16] E. Argyle. "Techniques for edge detection," Proc. IEEE, vol. 59, pp. 285-286, 1971.

[17] A. Yuille and T. A. Poggio . "Scaling theorems for zero crossings". IEEE Trans. Pattern Anal. Machine Intell.,vol. PAMI-8, no. 1, pp. 187-163, Jan. 1986.

TABLE 1. SIMULINK PROFILE REPORT: SUMMARY

Report generated 12-Dec-2011 22:33:35

Total recorded time:	1.70 s
Number of Internal Methods:	4
Clock precision:	0.00000004 s
Clock Speed:	2650 MHz
To write this data as VARIOUS_EDGE_DETECTION ProfileData in the base workspace	

Name	Time		Time/call	Self time	
sim	1.70312500	100.0%	1.70312500	0.00000000	0.0%
ModelInitialize	1.32812500	78.0%	1.32812500	1.32812500	78.0%
ModelExecute	0.23437500	13.8%	0.23437500	0.23437500	13.8%
ModelTerminate	0.14062500	8.3%	0.14062500	0.14062500	8.3%

Function List

TABLE 2. SIMULINK PROFILE REPORT: FUNCTION DETAILS

sim VARIOUS_EDGE_DETECTION
Time: 1.70312500 s (100.0%)
Calls: 1
Self time: 0.00000000 s (100.0%)

Function:	Time		Calls	Time/call
sim	1.70312500		1	1.70312500
Parent functions:				
none				
Child functions:				
ModelInitialize	1.32812500	78.0%	1	1.32812500
ModelExecute	0.23437500	13.8%	1	0.23437500
ModelTerminate	0.14062500	8.3%	1	0.14062500

TABLE 3. MODELINITIALIZE
VARIOUS_EDGE_DETECTION
Time: 1.32812500 s (78.0%)
Calls: 1
Self time: 1.32812500 s (78.0%)

Function:	Time	Calls	Time/call
ModelInitialize	1.32812500	1	1.32812500
Parent functions:			
sim		1	
Child functions:			
none			

TABLE 4. MODELEXECUTE
VARIOUS_EDGE_DETECTION
Time: 0.23437500 s (13.8%)
Calls: 1
Self time: 0.23437500 s (13.8%)

Function:	Time	Calls	Time/call
ModelExecute	0.23437500	1	0.23437500
Parent functions:			
sim		1	
Child functions:			
none			

TABLE 5. MODELTERMINATE
VARIOUS_EDGE_DETECTION
Time: 0.14062500 s (8.3%)
Calls: 1
Self time: 0.14062500 s (8.3%)

Function:	Time	Calls	Time/call
ModelTerminate	0.14062500	1	0.14062500
Parent functions:			
sim		1	
Child functions:			
none			