Vol 8. No. 3 – September, 2015

African Journal of Computing & ICT

© 2015 Afr J Comp & ICT – All Rights Reserved - ISSN 2006-1781
www.ajocict.net

Performance Evaluation of Image Edge Detection Techniques in the Domain of Human Computer Interaction

Adigun, A.A.
Department of Information and Communication Technology
Osun State University, Osogbo, Nigeria
Email: fempej2013@gmail.com

ABSTRACT

Edge detection is a kind of method of image segmentation based on range non-continuity. Image edge detection is one of the basal contents in the image processing and analysis, and also is a kind of issues which are unable to be resolved completely so far. Detecting edges is very useful in a number of contexts. It plays an important role in digital image processing and practical aspects of our life. This leads to the investigation of various edge detection techniques solved by the use of Time Variation Execution Method. With the comparative analysis, at the execution time, prewitt edge detector performs faster than the others, while canny edge detector produces better edges than the other edge detectors.

Keyword: Edge detection, Image, Prewitt edge detector, Canny edge detector, Time variation execution.

African Journal of Computing & ICT Reference Format:
Adigun, A.A. (2015): Performance Evaluation of Image Edge Detection Techniques in the Domain of Human Computer Interaction.
Afr J. of Comp & ICTs. Vol 8, No. 3. Pp 239-245. .

1. INTRODUCTION

Edge detection is a kind of method of image segmentation based on range non-continuity. Image edge detection is one of the basal contents in the image processing and analysis, and issues which are unable to be resolved completely. The separation of the image into object and background is a critical step in image interpretation. When we imitate the human visual system by using computer algorithms, quite a lot of problems can be encountered. When image is acquired, the factors such as the projection, mix, aberrance and noise are produced. These factors bring on image feature is blur, distortion and very difficult to extract image feature. This made it difficult to detect edge. [1][4]Detecting edges is the first step in the image segmentation and very useful in a number of contexts. Edge detection, feature extraction and object recognition heavily rely on the quality of the segmentation. Without a good segmentation algorithm, an object may never be recognizable. However, contours can be correctly reconstructed either by performing edge grouping or boundaries of segmented regions[2][3][6].

2. METHODOLOGY

Formulated Time Variation Execution method (TVEm) used to determine the faster performance of execution time and best image producer between edge detection techniques. TVEm works around differential operators to detect changes in the gradients of the grey levels. The component of differential operators comprises of Noise reduction by smoothing Noise contained in image through the input image I (i, j) with Gaussian filter as given by *F (i, j) = G*I (I, j), [5][7]*

Finding gradients used to detect the edges where the change in grayscale intensity is maximum. Required areas are determined with the help of gradient of images in i and j directions are given as

$$D_i = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \quad \text{And} \quad D_j = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

With smoothed image and giving gradients in i and j directions,

$$G_i = D_i * F(i,j) \quad \text{And} \quad G_j = D_j * F(i,j)$$

Therefore edge strength of gradient of a pixel is given by:

$$G = \sqrt{{G_i}^2 + {G_j}^2}$$

The direction of gradient is given by:

$$\theta = arctan\left(\frac{G_j}{G_i}\right)$$

,

Non maximum suppressions is carried out to preserves all local maxima in the gradient image, and deleting everything else this results in thin edges. For a pixel M (i, j) work round the gradient direction in the nearest 45°, then compare the gradient magnitude of the pixels in positive and negative gradient directions. If gradient direction is east then compare with gradient of the pixels in east and west directions say E (i, j) and W (i, j) respectively [9]][8.

Vol 8. No. 3 – September, 2015
African Journal of Computing & ICT
© 2015 Afr J Comp & ICT – All Rights Reserved - ISSN 2006-1781
www.ajocict.net

But, If the edge strength of pixel M (i, j) is larger than that of E (i, j) and W (i, j), then preserve the value of gradient and mark M (i, j) as edge pixel, if not then suppress or remove. Hysteresis Thresholding is the output of non-maxima suppression that still contains the local maxima created by noise. [11][13]A pixel M (i, j) having gradient strength G, following conditions exists to detect pixel as edge: If G < than t-low discard the edge, If G > than t-high keep the edge, If none of pixel (x, y)'s neighbours have high gradient strength but at least one falls between t-low and t-high search the 5 × 5 region to see if any of these pixels have a strnght greater than thigh. If so, keep the edge or Else, discard the edge.[10][14]

3. EDGE DETECTION CODES

```
function varargout = Edge_Detection_Techniques(varargin)
% EDGE_DETECTION_TECHNIQUES M-file for Edge_Detection_Techniques.fig
%      EDGE_DETECTION_TECHNIQUES, by itself, creates a new EDGE_DETECTION_TECHNIQUES or raises the existing
%      singleton*.
%
%      H = EDGE_DETECTION_TECHNIQUES returns the handle to a new EDGE_DETECTION_TECHNIQUES or the handle
to
%      the existing singleton*.
%
%
EDGE_DETECTION_TECHNIQUES('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in EDGE_DETECTION_TECHNIQUES.M with the given input arguments.
%


%      EDGE_DETECTION_TECHNIQUES('Property','Value',...) creates a new EDGE_DETECTION_TECHNIQUES or raises
the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Edge_Detection_Techniques_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Edge_Detection_Techniques_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Edge_Detection_Techniques

% Last Modified by GUIDE v2.5 25-May-2012 10:52:11

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Edge_Detection_Techniques_OpeningFcn, ...
                   'gui_OutputFcn',  @Edge_Detection_Techniques_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before Edge_Detection_Techniques is made visible.
function Edge_Detection_Techniques_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.

% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Edge_Detection_Techniques (see VARARGIN)

% Choose default command line output for Edge_Detection_Techniques
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Edge_Detection_Techniques wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Edge_Detection_Techniques_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Tstimg
% w=cd;


% cd(strcat(w,'\FingerprintDB'))
[Filename PathName]=uigetfile('*.jpg;*.bmp;*.tif','Select an Image');
% cd(w)
if Filename~=0
    Tstimg=[PathName,Filename];
    set(handles.text2,'string',Tstimg)
    axes(handles.axes1)
    imshow(Tstimg)
else
    msgbox('Select a Fingerprint')
    set(handles.text2,'string','')
end

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
img1=getimage(handles.axes1);
img2=rgb2gray(img1);
axes(handles.axes1);
imshow(img2)

% --- Executes on button press in pushbutton3.

function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Tstimg edgeopt
rgbimg=imread(Tstimg);
figure,imshow(rgbimg),title('original image')
%img2=histeq(rgbimg,64);
img2=rgbimg;
%% performing filtering
tic
h=fspecial('laplacian');
filteredimg=imfilter(img2,h);
h2=fspecial('gaussian');
smothenedimg=imfilter(rgbimg,h2);

level=graythresh(smothenedimg);

BW1=im2bw(rgbimg,level);
switch edgeopt
    case 'Prewitt'
%if edgeopt=='Prewitt'
edgeimg = edge(BW1,'prewitt');
type='Prewitt Method';
timehandles=handles.textp;
    case 'Sobel'
%elseif edgeopt=='Sobel'
edgeimg=edge(BW1,'sobel');
type='Sobel Method';
timehandles=handles.texts;
    case 'Canny'
%elseif edgeopt=='Canny'
edgeimg=edge(BW1,'canny');
type='Canny Method';
timehandles=handles.textc;
end
Eelapsed=toc;
axes(handles.axes3)
imshow(edgeimg)


set(timehandles,'string',Eelapsed)
%BW3comp=ones(size(BW1))-BW3;
BW4comp=ones(size(BW1))-edgeimg;
figure(1),imshow(smothenedimg),title('smothened image')

figure(2),imshow(filteredimg),title('Filterd image')

figure(3),imshow(BW1),title('Binary masking of the image')

figure(4),imshow(edgeimg),title(type)
figure(5),imshow(BW4comp),title('Edge compliment')
% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global edgeopt
 contents = cellstr(get(hObject,'String')); %returns popupmenu1 contents as cell array
 edgeopt=contents{get(hObject,'Value')}; % returns selected item from popupmenu1
set(handles.text4,'string',['Algorithm: ',edgeopt])

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close(gcf)
```

4. RESULTS

Edge detection methods investigated so far are further assessed by quality measures that give reliable statistical evidence to distinguish among the edges obtained. The absence of the ground true edge reveals the search for an alternative approach to assess and compare the quality of the edges resulted from the detectors exploited so far

The execution time for an image was documented for six months and the result were shown in the tables 1 and 2. The total averaged report for the execution time generated between April and September, 2014. The performance of the various edge detection techniques as shown in Fig..1.

Table 1: EVALUATED TABLE from April to September, 2014

MONTHS	PREWITT	SOBEL	CANNY
APRIL	27.63	29.54	80.43
MAY	28.21	30.02	81.80
JUNE	28.31	30.02	81.80
JULY	27.16	29.54	83.12
AUGUST	28.76	30.22	85.25
SEPTEMBER	28.65	30.02	82.82

The total execution time shown in Table 1 is used to derive the performance level of the best edge detection technique. The average total execution time shown in Table 2 with the corresponding graph plotted in Fig. 1.

TABLE 2: Total average of the execution time from Table 1.

EDGE TECHNIQUES	**PREWITT**	**SOBEL**	**CANNY**
AVERAGE TOTAL TIME	31.89	33.87	99.98

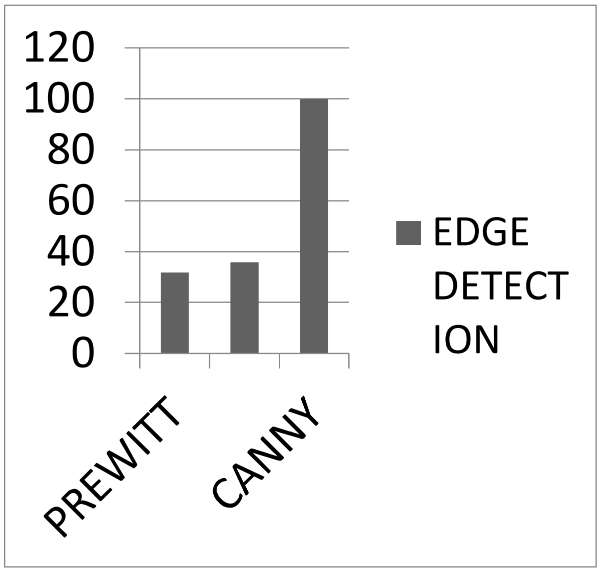

Fig. 1:Edge detection performance level

When compared the performance level and shortest execution time of edge detection techniques, Prewitt and Sobel showed the broken edges at some junctions. Though Prewitt worked faster than Sobel and Canny did, their results were of average quality compared to the Canny detectors. The Canny operator can detect the complete, continuous and detailed edges, but it also smoothens some edges in the smoothening process. Based on the speed, the canny is slowest operator in the particular application. The Sobel and Prewitt operators are similar based on the speed. The Sobel operators can save most of the high frequency information of the image, but the Prewitt operators only save a little high - Frequency information

5. CONCLUSION

This study has provided the best levels of performance and shortest execution time with image of human computer interaction. The study demonstrated that Prewitt edge detector performs faster than the canny and sobel techniques but Canny edge detector gives better result than others with some positive points. It is less sensitive to noise, adaptive in nature, resolved the problem of converting to a grayscale image first, provides good localization and detects sharper edges as compared to others..

REFERENCES

[1] J. F. Canny. (2008) "A computational approach to edge detection". IEEE Trans. Pattern Anal. Machine Intell., vol.PAMI-8, no. 6, pp. 679-697, Journal of Image Processing (IJIP), Volume (3) : Issue (1)

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

[3] D. Marr and E.Hildreth.(2008) "Theory of Edge Detection". Proceedings of the Royal Society of London. Series B, Biological Sciences,, Vol. 207, No. 1167, pp. 187-217.

[4] M. Heath, S. Sarkar, T. Sanocki, and K.W. Bowyer (1997). "A Robust Visual Method for Assessing the Relative Performance of Edge Detection Algorithms". IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 12, pp. 1338-1359, Dec. 1997

[5] V. Lacroix (1998). ªA Three-Module Strategy for Edge Detection,ºIEEE Trans. Pattern Analysis Machine Intelligence, vol. 10, pp. 803-810, 1998.

[6] Z.D. Lan and R. Mohr (1998). ªDirect Linear Sub-Pixel Correlation by Incorporation of Neighbor Pixels Information and Robust Estima-tion of Window Transformation,º Machine Vision Applications,vol. 10, pp. 256-268, 1998.

[7] R. Lenz (1995). ªInvestigation of Receptive Fields Using Representations of the Dihedral Groups,ºJ. Visual Comm. and Image Representation, vol. 6, pp. 209-227, 1995.

[8] P.Meer, S.Wang, and H.Wechsler (1989). ªEdge Detection by Associative Mapping,ºPattern Recognition,vol. 22, pp. 491-503, 1989.

[9] P. Meer and I. Weiss (1992)., ªSmoothed Differentiation Filters for Images,º J. Visual Comm. and Image Representation, vol. 3, pp. 58-72, 1992.

[10] M. Shin, D. Goldgof, and K.W. Bowyer (1998). ªAn Objective Compar-ison Methodology of Edge Detection Algorithms for Structure from Motion Task,º Empirical Evaluation Techniques in Computer Vision,K.W. Bowyer and P.J. Phillips, eds., IEEE CS Press, pp. 235-254, 1998.

[11] Paulus (2007). Color image processing: methods and applications in color image Segmentation Selected Techniques, chapter 5, pp.103-128, CRC Press, Boca Rato, Fla, USA, 2007.

[12] Maher I.RAJAB AND KHALID A.AL-HINDI (2008). Analysis of Neural Network Edge Pattern Detectors in Terms of Domain Functions, SEAS Transactions on Information Science & Applications ",issue 2, Volume 5, February 2008.

[13] A.Jiang,C.L.Chuang,Y.L.Lu and C.S.Fahn (2007). Mathematical-morphology-based edge detectors for detection of thin edges in low-contrast regions in The Institution of Engineering and Technology (IET) Image Processing,pp.269-277,2007.

[14] J. M. Niya, A. Aghagolzadeh , M. A. Tinati, and S.Feizi (1998). 2 -step wavelet-based edge detection using gabor and cauchy directional wavelets, IEEE Trans. Acoustics,Speech, Signal Processing, vol. 37, page(s): 2091-2110, December 1989.