IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 2, February 2016

Image Intensification using Various Edge Detection Mechanisms

Reena Jangra[1], Abhishek Bhatnagar[2]

M.Tech Student in C.E, IIET (Jind) under KUK, India[1]

Asstt. Prof. in C.S.E, IIET (Jind) under KUK, India[2]

Abstract: Edge (Boundary line)s are borders between various textures. Edge (Boundary line) can be defined as discontinuities in image(picture) intensity from one pixel to another. Edge (Boundary line) s for an image (picture) are always essential character that suggest an pointer for a better-quality frequency. Detection of Edge (Boundary line)s for an image(picture) may help for image(picture) segmentation, data compression, & also help for well identical, such as image(picture) modernization & so on. There are many way to make Edge (Boundary line) detection. Most universal method for Edge (Boundary line) detection is to analyze discrimination of an image(picture). Edge (Boundary line) detection is an image(picture) processing technique for finding borders of objects within image(picture)s. It works by detecting discontinuities in brightness. Edge (Boundary line) detection is used for image(picture) segmentation & data extraction in areas such as image(picture) processing, computer vision, & machine vision. Common Edge (Boundary line) detection algorithms include Sobel, Canny, Roberts, Prewitt & fuzzy logic methods.

Keywords: Edge (Boundary line) detection, Canny Edge (Boundary line) detection, Sobel operator.

INTRODUCTION

Edge (Boundary line) detection is a set of mathematical methods which aim at identify points in a digital image(picture) at which image(picture) intensity changes sharply or, more formally, has discontinuities.

Points at which image(picture) intensity changes sharply are typically organized into a set of curved line segments termed Edge (Boundary line)s.

Same problem of verdict discontinuities in 1D signals is known as step detection & problem of verdict signal discontinuities over time is known as change detection.

Edge (Boundary line) detection is a fundamental tool in image(picture) processing, machine vision & computer vision, particularly in areas of feature detection & feature extraction.

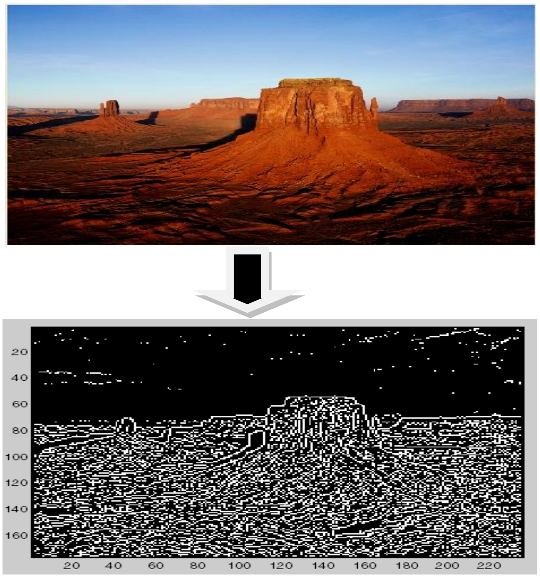

Fig. 1[Detection of highlighted points]

EDGE (BOUNDRY LINE) PROPERTIES

The Edge (Boundary line)s extracted from a two-dimensional image(picture) of a three-dimensional scene can be classify as either:

- viewpoint dependent
- viewpoint independent.

Viewpoint dependent Edge (Boundary line) may change as viewpoint changes, & typically reflects geometry of scene, such as objects occluding one another.

Viewpoint independent Edge (Boundary line) typically reflects inherent properties of three-dimensional objects, such as surface markings & surface shape.

A typical Edge (Boundary line) might for instance be border between a block of red color & a block of yellow.

For a line, there can therefore usually be one Edge (Boundary line) on each side of line.

PROBLEM FORMULATION

Edge (Boundary line) detection is a basic tool used in image(picture) processing, basically for characteristic detection & extraction, which aim to categorize points in a digital image(picture) where intensity of image(picture) changes sharply & find discontinuities.

Purpose of Edge (Boundary line) detection is considerably reducing amount of data in an image(picture) & preserves structural properties for additional image(picture) processing.

In a grey level image(picture) Edge (Boundary line) is a local feature that, with in a neighborhood separates the regions in each of which gray level is more or less uniform with in various values on two sides of Edge (Boundary line).

For a noisy image(picture) it is difficult to detect Edge (Boundary line)s as both Edge (Boundary line) & noise contains high frequency contents which results in blurred & unclear result.

DIFFERENT EDGE (BOUNDRY LINE) DETECTION METHODOLOGIES

Edge (Boundary line) detection makes use of variousial operators to distinguish changes in gradients of grey levels. It is divided into two main categories.

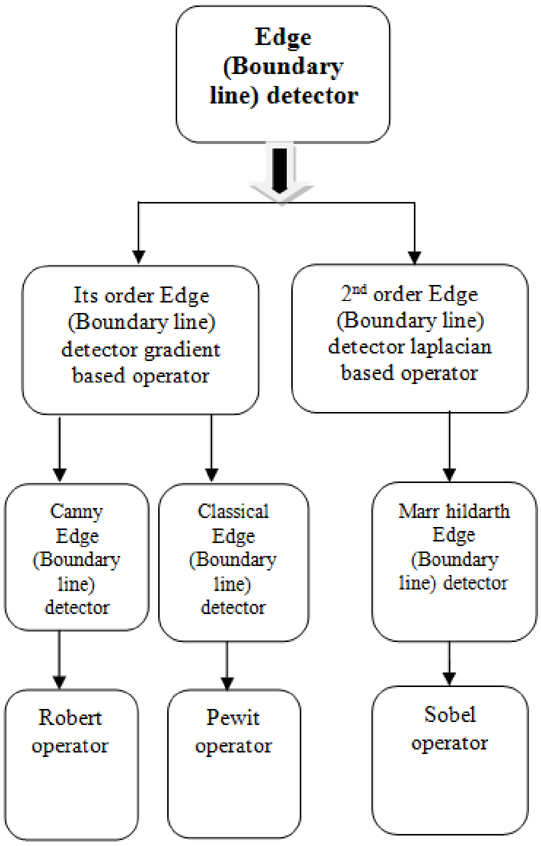
Edge (Boundary line) detector
Its order Edge (Boundary line) detector gradient based operator
2nd order Edge (Boundary line) detector laplacian based operator
Canny Edge (Boundary line) detector
Classical Edge (Boundary line) detector
Marr hildarth Edge (Boundary line) detector
Robert operator
Pewit operator
Sobel operator

Fig2. Flowchart of Classical operators

OBJECTIVES

Canny Edge (Boundary line) detector have highly developed algorithm derived from previous work of Marr & Hildreth.

It is an best possible Edge (Boundary line) detection technique as provide good detection, clear response & good localization.

It is widely used in current image(picture) processing techniques with further improvements. Objective of research is to High light benefit of canny Edge (Boundary line) detection over usual Edge (Boundary line) detection schemes.

On analyzing all these Edge (Boundary line) detection techniques , it is found that canny gives best Edge (Boundary line) detection.

Following are some points throwing light on advantages of canny Edge (Boundary line) detector as compared to other detectors discussed in this paper:

1. Less Sensitive to noise: As compared to classical operators like Prewitt, Robert & Sobel canny Edge (Boundary line) detector is less sensitive to noise. Its uses Gaussian filter that removes noise at a great level as compared to above noise filters.

2. Adaptive in nature: Classical operator have fixed kernels so cannot be adapted to a given image(picture). While performance of canny algorithm depends on variable or adjustable.

3. Remove streaking problem: Classical operators' like Robert uses single thresholding technique but it results into streaking.

Streaking means, if Edge (Boundary line) gradient just above & adjust below set threshold limit it removes useful part of connected Edge (Boundary line), & leave disconnected final Edge (Boundary line).

To overcome from this drawback canny detector uses 'hysteresis' technique which uses two threshold values _789 & _5_65 as discussed above in canny algorithm.

METHODOLOGY/ PLANNING OF WORK

Our plan is to use MATLAB programming software as a tool for developing this Biomedical Image(picture) Processing software package.

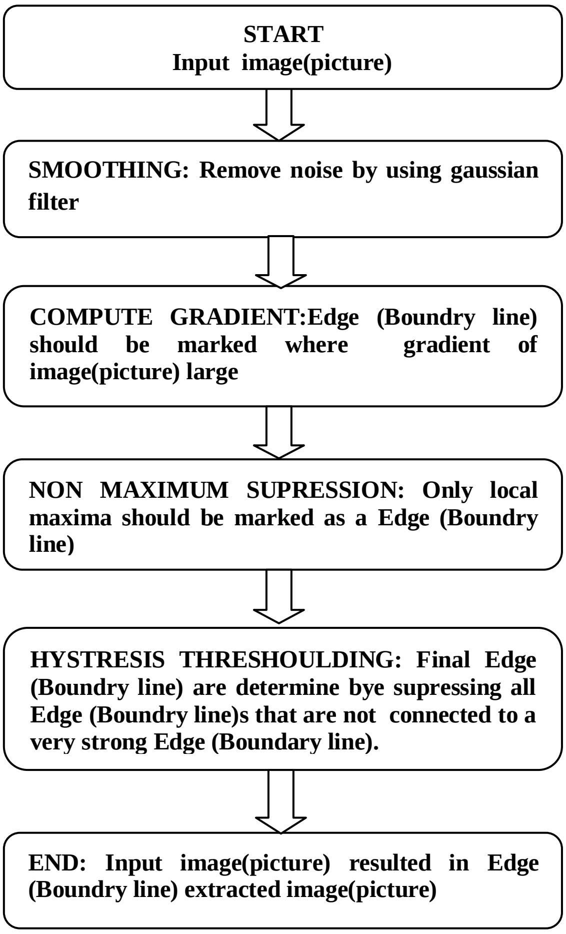
START
Input image(picture)
SMOOTHING: Remove noise by using gaussian filter
COMPUTE GRADIENT:Edge (Boundry line) should be marked where gradient of image(picture) large
NON MAXIMUM SUPRESSION: Only local maxima should be marked as a Edge (Boundry line)
HYSTRESIS THRESHOULDING: Final Edge (Boundry line) are determine bye supressing all Edge (Boundry line)s that are not connected to a very strong Edge (Boundary line).
END: Input image(picture) resulted in Edge (Boundry line) extracted image(picture)

Fig 3. Flowchart of canny Edge (Boundary line) detection

DOI 10.17148/IJARCCE.2016.5209

Such techniques (image(picture) improvement, filtering, segmentation & morphological operation) in Biomedical Image(picture) Processing.

This software package should capable to display input image(picture), output image(picture) & various click button for various image(picture) processing techniques as well as description of techniques so students will learn effectively application on how biomedical image(picture) is analyzed instead of learning mathematical algorithm for such techniques.

A number of software packages for image(picture) processing & editing have increased over years. various steps of image(picture) processing make each of software packages differ with various relative strengths.

Having accurate software & suitable processing techniques is necessary to assurance reliability of data processing.

IMPLEMENTATION

Code of canny in mat lab

```
function [sFinal,thresh] = canimp(image1, mLow, mHigh, sig)
if (nargi < 1)
  error(' Need a NxN image matrix');
elseif (nargi ==1)
  mLow = 0.5; mHigh = 2.5; sig = 1;
elseif (nargi == 2)
  mHigh = 2.5; sig = 1;
elseif (nargi == 3)
  sig = 1;
end
origImage = image1;
if (ndims(image1)==3)
  image1 =double(rgb2gray(image1));
end

% DERIVATIVE OF GAUSSIAN
dG=dgauss(sig1);
[dummy, filterLen] = size(dG);
offset = (filterLen-1)/2;
sy = conv2(image1, dG ,'same');
sx = conv2(image1, dG','same');
[m, n]=size(image1);
sx = sx(offset+1:m-offset, offset+1:n-offset);
sy = sy(offset+1:m-offset, offset+1:n-offset);
sNor = sqrt( sx.^2 + sy.^2 );
sAngle = atan2( sy, sx) * (180.0/pi);
sx(sx==0) = 1e-10;
sSlope = abs(sy ./ sx);
sAorig = sAngle;
y = sAngle < 0;
sAngle = sAngle + 180*y;
        binDist =    [-inf 45 90 135 inf];
[dummy, b] = histc(sAngle,binDist);
sDiscreteAngles = b;
[m,n] = size(sDiscreteAngles);
        sDiscreteAngles(1,:) = 0;
sDiscreteAngles(end,:)=0;
sDiscreteAngles(:,1) = 0;
sDiscreteAngles(:,end) = 0;
sEdge (Boundry line)points = zeros(m,n);
sFinal = sEdgepoints;
lowT  = mLow * mean(sNor(:));
highT = mHigh * lowT;
thresh = [ lowT highT];
 gradDirn = 1;
idxs = find(sDiscreteAngles ==  gradDirn);
slp = sSlope(idxs);
gDiff1 = slp.*(sNor(idxs)-sNor(idxs+m+1)) + (1-slp).*(sNor(idxs)-sNor(idxs+1));
gDiff2 = slp.*(sNor(idxs)-sNor(idxs-m-1)) + (1-slp).*(sNor(idxs)-sNor(idxs-1));
okIdxs = idxs( gDiff1 >=0 & gDiff2 >= 0);
sEdge points(okIdxs) = 1;
 gradDirn = 2;
idxs = find(sDiscreteAngles ==  gradDirn);
inSp = 1 ./ sSlope(idxs);
gDiff1 =   inSp.*(sNor(idxs)-sNor(idxs+m+1)) + (1-inSp).*(sNor(idxs)-sNor(idxs+m));
gDiff2 =   inSp.*(sNor(idxs)-sNor(idxs-m-1)) + (1-inSp).*(sNor(idxs)-sNor(idxs-m));
okIdxs = idxs( gDiff1 >=0 & gDiff2 >= 0);

sEdge points(okIdxs) = 1;
 gradDirn = 3;
idxs = find(sDiscreteAngles ==  gradDirn);
inSp = 1 ./ sSlope(idxs);
gDiff1 =   inSp.*(sNor(idxs)-sNor(idxs+m-1)) + (1-inSp).*(sNor(idxs)-sNor(idxs+m));
gDiff2 =   inSp.*(sNor(idxs)-sNor(idxs-m+1)) + (1-inSp).*(sNor(idxs)-sNor(idxs-m));

okIdxs = idxs( gDiff1 >=0 & gDiff2 >= 0);
sEdge points(okIdxs) = 1;
 gradDirn = 4;
idxs = find(sDiscreteAngles ==  gradDirn);
slp = sSlope(idxs);
gDiff1 = slp.*(sNor(idxs)-sNor(idxs+m-1)) + (1-slp).*(sNor(idxs)-sNor(idxs-1));
gDiff2 = slp.*(sNor(idxs)-sNor(idxs-m+1)) + (1-slp).*(sNor(idxs)-sNor(idxs+1));

okIdxs = idxs( gDiff1 >=0 & gDiff2 >= 0);
sEdge points(okIdxs) = 1;
sEdge points = sEdge points*0.6;
x = find(sEdge points > 0 & sNor < lowT);
sEdge points(x)=0;

x = find(sEdge points > 0 & sNor  >= highT);
sEdge points(x)=1;
oldx = [];
            x = find(sEdge points==1);
while (size(oldx,1) ~= size(x,1))
  oldx = x;
  v = [x+m+1, x+m, x+m-1, x-1, x-m-1, x-m, x-m+1, x+1];
  sEdge points(v) = 0.4 + sEdge points(v);

  y = find(sEdge points==0.4);
  sEdge points(y) = 0;
  y = find(sEdge points>=1);
  sEdge points(y)=1;
  x = find(sEdge points==1);
end
x = find(sEdge points==1);
sFinal(x)=1;

figure(1);
imagesc(sFinal); colormap(gray); axis image;
```

Running canimp code

```
>>x=imread('Tulips.jpg')
>>imagesc(x)
```

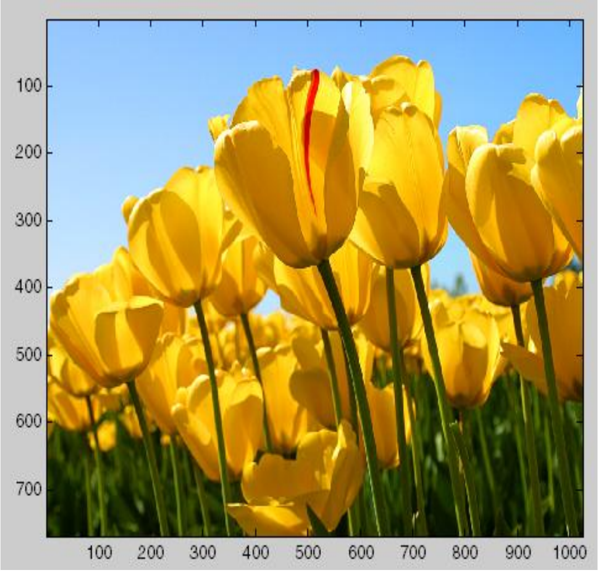

Fig. 4: original picture

```
>> canny(x,.5,.5,.5)
```

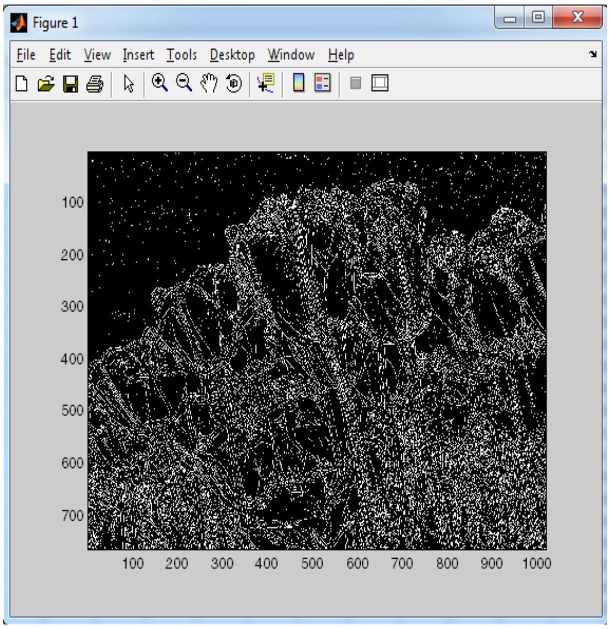

Fig. 5: After applying canny

APPLICATIONS

It finds practical application in Runway Detection & Tracking for Unmanned Aerial Vehicle, in brain MRI image(picture), cable insulation layer measurement, Real-time facial expression recognition, Edge (Boundry line) detection of river regime, Automatic Multiple Faces Tracking & Detection.

Canny Edge (Boundry line) detection technique is used in license plate reorganization system which is an important part of intelligent traffic system (ITS), finds practical application in traffic management, public safety & military department. It also finds application in medical field as in ultrasound, x –rays etc.

FUTURE SCOPE & CONCLUSION

In this research we have studied & evaluate various Edge (Boundry line) detection techniques. We have seen that canny Edge (Boundry line) detector gives enhanced result as compared to others with some positive points.

It is less sensitive to noise, adaptive in nature, resolved problem of streaking, gives good localization & detects sharper Edge (Boundry line) as compared to others.

It is consider as best possible Edge (Boundry line) detection technique hence lot of work & enhancement on this algorithm has been done & additional improvements are possible in future as an improved canny algorithm can detect Edge (Boundry line)s in color image(picture) without converting in gray image(picture), improved canny algorithm for automatic extraction of moving object in image(picture) supervision.

REFERENCES

1. The Technology of Night Vision by Harry P. Montoro, ITT Night Vision http://www.photonics.com/EDU/Handbook.aspx?AID=25144
2. A. Marion *An Introduction to Image Processing*, Chapman & Hall, 1991
3. Azeema Sultana, Dr. M. Meenakshi, "Design & Development of FPGA based Adaptive Thresholder for Image Processing Applications" ,on line access
4. Gerhard X. Ritter; Joseph N. Wilson, " Handbook of Computer Vision Algorithms in Image Algebra" CRC Press, CRC Press LLC ISBN:0849326362 Pub Date: 05/01/96
5. N. Nacereddine, L. Hamami, M. Tridi, & N. Oucief , "Non-Parametric Histogram-Based Thresholding Methods for Weld Defect Detection in Radiography " ,online access.
6. Otsu,N., "A Threshold Selection Method from Gray-Level istograms,"IEEE Transactions on Systems, Man, & Cybernetics, Vol. 9, No. 1, 1979, pp. 62-66.
7. Elham Ashari , Richard Hornsey, " FPGA Implementation of Real-Time Adaptive Image(picture) Thresholding" ,online access
8. http://en.wikipedia.org/wiki/Digital_image(picture)_processing
9. J. Canny, "A Computational Approach to Edge Detection," IEEE Trans. Pattern Analysis & Machine Intelligence, PAMI-8, 6, November 1986, 679–698.

DOI 10.17148/IJARCCE.2016.5209