Segmentation using Edge Detection Algorithm to Analyze Internal Features of a Digital Image

Rahul Kumar Singh¹, Shraddha Masis²
¹M.Tech, ²Reader, School of Computer Science & IT, DAVV, Indore, India

Abstract—Segmentation is a vital field of image processing which deals with the recognition and identification of objects of interest in an image. By segmenting the object of interest, further research can be carried out to reveal significant details of the objects. There are various methods of segmentation like point detection, edge detection and line detection. From the mentioned methods, edge detection has an upper hand over other methods. However, it depends on the problem to be analyzed and based on this, different detectors work in a different way. This means that edge detection operator although suitable to most problematic situation is not free from error. This report will deal with edge detection algorithms and highlight what lay ahead and behind these. Edge of an image gives a blueprint of how the image is going to look like. Particularly the useful and important details of an image can be traced by detecting edges. However, not every edge is detected; it depends on the image intensity and noise that is present in the image. This work and report is about analysis and comparison of images that can be done by tracing edges. This work is about an application which takes two images as inputs. The images contain closed figures of different types. Here, several image processing operations are applied to reveal some interesting facts about the images. Special emphasis has been put on the enclosed figures that are the object of interest. First, the object of interests has been indentified and then the significant properties have been extracted and analyzed to compare whether the images are similar. For this, segmentation of the object of interest is very vital and based on the accuracy of segmentation, results are found.

Keywords—Image Processing, Edge Detection, Segmentation.

I. INTRODUCTION

An image is a two dimensional matrix where x and y are its spatial coordinate and f is the amplitude at any point(x, y). Such an image is called the intensity or gray level of the image at that point. When its spatial coordinates and amplitude are all finite and discrete quantities, the image is defined as a digital image. Images can be either of these types-binary, grayscale, indexed and RGB.

Image Processing is a wide field that encompasses image analysis and computer vision. Here, the input is an image but the output is attributes extracted from the image.

An application has been developed in which two images are taken as input. The aim of the application is to prove that images varying in size and orientation are similar. In most cases, the level used is high level where the input are attributes extracted from images but the output is the understanding that can be drawn from it.

Problem Situation: In a large database containing innumerable images, comparison based on images varying in size and orientation could eliminate redundancy as images falling in both the category would be similar and data related to them can be stored in any of them instead of storing in all of them.

Benefits: Such comparison method will help in reducing redundant data meaning optimization of time, budget and resources.

II. OBJECTIVES

1. Extract properties of image
2. Extract properties of object of interest
3. Compare images
4. Figures differing in size and orientation should be similar

III. LITERATURE SURVEY

Canny operator is the most powerful of all existing operator. The existing status shows that there are comparison algorithms for images as used in face recognition system which falls under the domain of computer vision. Such algorithms although very complex are very efficient. However, any comparison algorithms under the domain of image processing hardly exist.

IV. SEGMENTATION

A. Scope

Segmentation is a vital field of image processing where an image is segmented to reveal crucial details. Image segmentation is mainly divided into two types depending on the behavior of the images. Detecting Discontinuities (contextual): It means to divide an image based on abrupt changes in intensity which includes image segmentation algorithms like edge detection.
Detecting similarities (non-contextual): It means to divide an image into regions that are same with respect to a set of predefined criterion.

Moreover, it is classified into three types i.e. Edge Detection Based, Region Based and Spatial Based Segmentation.

Segmentation can be used by any of these methods-point, line and edge detection. Edge detection having an edge over other segmentation is found suitable in this scenario.

B. Need for Segmentation

Segmentation is done to extract features of object of interest. For comparison results, object of interest needs to be identified and image processing operations have to be applied on them to get the estimated output.

C. Edge detection algorithms

Edge detection technique is one of the best ways to segment images based on their edges. Edge detection is one of the first steps of feature extraction of an image. Edge detection characterizes the boundaries in the image and thus reduces the amount of data, filters out the unnecessary information from the image and provides the structural properties of the image. Various types of edge detectors are there viz sobel, prewitt, Robert, Laplacian and Canny. Canny being the most powerful is used as an edge detection operator.

V. METHODOLOGY

a) Images of various types viz RGB, binary, indexed and grayscale are taken as input: Images include images of all formats viz png, jpeg, tiff.

b) Edges are detected using canny operator: Canny being the most powerful is used to detect edges by using function edge.

c) Edges are displayed: On clicking the ‘Detect Edges’, the function edge is evoked and the result is displayed on the axes.

d) Compare button is clicked: On clicking the ‘compare’ button, the callback to the button is evoked.

e) Image is segmented to identify objects of interest: The region of interest is identified upon which various operation will be based

f) Image processing operations are applied on the object of interest: Image processing operations have been applied on the objects to get the parameters which will determine the comparison result.

g) Compute results of Image processing on the object of interest: After getting the results, these are tested to verify if the desired results have been found.

h) The result is displayed: The result is made visible on the display chart.

VI. IMPLEMENTATION

Methodology has been carried out on MATLAB 2009b.

Problems encountered

Segmenting the object from image and calculating the area: In several cases, the area found is of the entire image and not of the object specifically. Thus, misleading results have been found due to inaccuracy in segmentation.

Detecting the corners of a object: This will give the vertices of the objects such as square, triangle etc which will give the length of the sides of such objects. The two objects will be compared with their sides. However, unless accurate corners are detected, it produces misleading results.

For indexed image to be compared the image is first converted to RGB type and the same operation is done. Converting indexed to RGB means losing some data of the original image which needs to be compromised.

Both the images need to be brought to same scale to enable comparison. This means image with high dimension must be scaled down to match with image with low dimension. Some of the functions which didn’t yield results-

B = BBOUNDARIES(BW) traces the exterior boundary of objects along with boundaries of holes inside these objects. It also penetrates into the outermost objects (parents) and traces their children (objects completely enclosed by the parents). BW is a binary image where nonzero pixels constitute the object and 0-pixels constitute the background. B is a P-by-1 cell array, where P is the number of objects and holes. Every cell contains a Q-by-2 matrix, where Q being the number of boundary pixels for the corresponding region. Each row of these Q-by-2 matrices contains the row and column coordinates of a boundary pixel. The coordinates are ordered in a clockwise direction.

- bweuler: BWEULER Euler number of binary image

EUL = BWEULER(BW,N) returns the Euler number for the binary image BW. EUL is a scalar value obtained by subtracting the total number of holes in those objects from the total number of objects in the image. N can be either 4 or 8, where 4 specifies 4-connected objects and 8 specifies 8-connected objects; if the argument is omitted, it defaults to 8. After few experiments, it didn’t show desired result.
• **bwtraceboundary**: It traces object in binary image

\[ B = \text{BWTRACEBOUNDARY}(BW,P,FSTEP) \]

traces the boundary of an object in a binary image \( BW \), in which nonzero pixels constitute the object and 0-pixels belong to the background. \( P \) is a two-element vector containing the row and column coordinates of the initial point on the object boundary. \( FSTEP \) is a string specifying the initial search direction for the next object pixel connected to \( P \). \( FSTEP \) can be either of the following strings: ‘N’, ‘NE’, ‘E’, ‘SE’, ‘S’, ‘SW’, ‘W’, ‘NW’, where N is north, NE stands for northeast, etc. \( B \) is a Q-by-2 matrix, where Q is the number of boundary pixels for the region. \( B \) contains the row and column coordinates of the boundary pixels. In several experiments, it didn’t show desired result.

• **bwlabel**: It contains label connected components in 2-D binary image

\[ L = \text{BWLABEL}(BW,N) \]

gives an output matrix \( L \), of the same size as \( BW \), containing labels for the connected components in BW. The elements of \( L \) are integer values greater than or equal to 0. The pixels labeled 0 are the background. The pixels labeled 1 constitute the 1st object, the pixels labeled 2 make up a 2nd object, and so on.

A. **Problems overcome**

Edge detection is done through the most powerful operator named Canny. Then imfill operation has been done followed by imsubtract to segment the area of interest.

B. **Important functions that showed desired results**

• **imfill**: Imfill fills image regions and holes.

\[ BW = \text{IMFILL}(BW1,\text{LOCATIONS}) \]

conducts a flood-fill operation on background pixels of the input image \( BW1 \). It starts from the points specified in LOCATIONS. LOCATIONS can be of type P-by-1 vector, in such case it contains the linear indices of the starting locations. LOCATIONS can also be a P-by-NDIMS(IM1) matrix, in which case each row contains the array indices of one of the starting locations.

• **bwarea**: Calculates area of objects in binary image.

\[ \text{TOTAL} = \text{BAREA}(BW) \]

estimates the area of the objects in binary image \( BW \). \( \text{TOTAL} \) is a scalar output whose value relates roughly to the total number of "on" pixels in the image. However, it may not be exactly the same because different patterns of pixels are weighted differently.

In an image which has a background and certain objects, the function can’t distinguish between objects and background. Thus, the area obtained is of the whole image and not of the object specifically.

- regionprops(area,centroid,filledarea,perimeter): Measure properties of image regions.
- Imsubtract(fill1,ed) finds the difference between two images.

This function can be used to calculate properties such as area, centroid, perimeter of any image. Prior to this, the object of interest has to be identified. In order to calculate area, first edge detection algorithm has been applied. Next, imfill performs flood fill operation on zero pixel. After that, image subtraction is done between filled portion of the figure and the earlier figure which produces the area required.

**ALGORITHM**

**For figures varying in Rotation:**

(edge1=edge(bw1,'canny'));

(fill1=imfill(pass,'holes'));

(sub1=imsubtract(fill1,pass));

(area1=bwarea(sub1));

(edge2=edge(bw2,'canny'));

(fill2=imfill(pass2,'holes'));

(sub2=imsubtract(fill2,pass2));

(area2=bwarea(sub2));

if area1==area2

set(handles.text4,'string','The images are similar');

elseif ((area1-area2)<=1) & (area1-area2)>=-1

set(handles.text4,'string','The images are similar but rotated');

else

set(handles.text4,'string','The images are dissimilar');

end

**For figures varying in size:**

re=regionprops(bw1,'centroid');
centroid=cat(1,re.Centroid);
axes(handles.Firstaxes);
hold on
plot(centroid(:,1), centroid(:,2), 'b*')
hold off;
re2=regionprops(bw2,'centroid');
centroid2nd=cat(1,re2.Centroid);
axes(handles.Secondaxes);
hold on
plot(centroid2nd(:,1), centroid2nd(:,2), 'b*')
hold off;
re=regionprops(bw1,'Perimeter');
re2=regionprops(bw2,'Perimeter');
if centroid==centroid2nd
    set(handles.text4,'string','The images are similar');
else
    set(handles.text4,'string','The images are not similar');
end
VII. CONCLUSION

Image analysis and image processing share almost a common platform and scope for dealing with images. Image Analysis is however a very subjective area of work which depends on image processing results. While using several edge detectors which have different pros and cons for different images, it’s vital to analyze internal features of image through which attributes can be extracted. Edge detectors can give hint of what kinds of images are likely to detect edges. Several operators have been applied on different images and compared with each other. Canny operator, being the most powerful of all existing edge detectors has been used to detect edges. Segmentation using various functions such as imfill, regionprops, blabel have been used to identify object of interest.

Getting accurate segmentation result is the main hurdle and once the object has been identified, parameters such as centre of mass, area, perimeter have been applied to compute result. A plethora of options have been provided to the user to compare. The project is concerned with analyzing features by detecting edges to check if any pattern can be discovered. The application developed is able to distinguish between similar images and similar images with rotation.

VIII. FUTURE WORK

Addition of graphical representation can enhance understanding of similarity/dissimilarity.

REFERENCES