CHAPTER 5

SEGMENTATION USING EDGE FLOW VECTOR

5.1 INTRODUCTION

Image segmentation is one of the widely studied problems in image processing, and has found its application directly or indirectly in task, such as object detection, object tracking and recognition, content-based image retrieval, and medical image analysis. There have been many techniques proposed to address the problem of recognizing and tracking people in video data. Many of the limitations stem from the use of background subtraction algorithms, binary blob detection and relatively crude features, meaning that it is desirable to see the whole of the person being tracked. The various segmentation methods are Thresholding, Clustering methods, Edge detection, Region-growing methods, Split-and-merge methods, Watershed transformation, Model based segmentation, Multi-scale segmentation etc..

Segmentation evaluation is concerned with measuring and comparing the performance and characteristics of segmentation algorithms. The objective is to measure and compare attributes of segmentation algorithms that are likely to be pertinent to a wide range of applications. If a segmentation evaluation technique can demonstrate that one segmentation algorithm is significantly better than some others, for a given set of assumptions, then a systems designer can reasonably exclude the other algorithms from further consideration, provided the assumptions hold good for their particular application. If the evaluation is sufficiently generic, then
evaluating the existing segmentation algorithms can potentially make designing a whole range of systems significantly easier. Research efforts in segmentation can then focus on optimizing, modifying, or generalizing techniques shown to be effective for a wide range of applications.

5.2 EDGE DETECTION FOR IMAGE SEGMENTATION

The strength of many image processing and computer vision problems depends on the spotting of the meaningful edges. Edge detection refers to the process of identifying and locating sharp discontinuities in an image.

It becomes more intriguing when colour images are considered because of their multi-dimensional nature. Colour images put up exact information about the object, which will be more useful for further operations than gray scale images. Due to some inescapable reasons such as deformation, long suit variation, noise, segmentation errors, convergence, and occlusion of objects in images, it is usually inconceivable to extract complete object contours or to segment the whole objects. Due to a deficiency of object edge information, the output image is not visually agreeable. A large number of methods are available in the literary works to segment images. This task is difficult and very important, since the output of an image segmentation algorithm can be fed as input to higher-level processing tasks.

Most edge detection methods work on the assumption that an edge occurs where there is a discontinuity in the intensity function or a very steep intensity gradient in the image. The discontinuities are abrupt changes in pixel intensity, which characterize the boundaries of objects in a scene. In typical images, edges characterize the object boundaries, and are therefore useful for segmentation, registration and identification of objects in a scene. There are
many techniques used for edge detection; some of them are Canny edge detection, Marr–Hildreth algorithm, Sobel Operator, Prewitt, LoG.

In Canny edge detection, detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. The Marr–Hildreth algorithm is a method of detecting the edges in digital images that is in the form of continuous curves, where there are strong and rapid variations in the image brightness. The Marr–Hildreth edge detection method is simple and operates by convolving the image with the LoG function, or, as a fast approximation by the difference of Gaussians. Then, zero crossings are detected in the filtered result to obtain the edges. The two main limitations are that it generates responses that do not correspond to edges, the so-called "false edges", and the localization error may be severe at the curved edges.

The Sobel operator is used in image processing, particularly within edge detection algorithms. Technically, it is a discrete differentiation operator, computing an approximation of the gradient of the image intensity function. At each point in the image, the result of the Sobel operator is either the corresponding gradient vector or the norm of this vector. The Sobel operator is based on convolving the image with a small, separable, and integer valued filter in the horizontal and vertical direction, and is therefore relatively inexpensive in terms of computations.

Zero Crossing uses the second derivative, and includes the Laplacian operator. It has fixed characteristics in all directions, and is sensitive to noise. Zero-crossing is used for the second directional derivative of the image intensity function. The Gaussian filtering is combined with the Laplacian to break down the image, where the intensity varies, to detect the edges effectively. These edge detection operators can have a better edge effect under the circumstances of the obvious edge and low noise. But the actual collected
image has lots of noises. So, many noises may be considered as the edges to be detected.

There are many methods for edge detection, but most of them can be grouped into two categories: search-based and zero-crossing based. The search-based methods detect edges by first computing a measure of the edge strength, usually a first-order derivative expression such as the gradient magnitude, and then searching for local directional maxima of the gradient magnitude using a computed estimate of the local orientation of the edge, usually in the gradient direction. The zero-crossing based methods search for zero crossings in a second-order derivative expression, computed from the image in order to find the edges. As a pre-processing step to edge detection, a smoothing stage, typically Gaussian smoothing, is almost always applied.

**Observation from various Methods**

From the above discussion, the characteristics of the various edge detection methods are observed and summarized as follows:

- Changes in the lighting conditions
- The background is dynamic
- Luminance and geometrical features
- Sensitivity to noise
- Missing true edges
- Detecting an edge where it does not exist (false edge)
- Position of the detected edge to be shifted from its true location (shifted edge or dislocated edge).
- Producing thin or thick lines
- Inaccurate
In this work, it is decided to apply the enhanced edge flow method. The major contributions of the improved edge flow vector detection systems are as follows:

- An effective segmentation technique based on an edge field computed directly from the images.
- The flow field can be computed from various image features, including colour, texture and intensity edges.
- A new edge function that is more precise than the commonly used gradient magnitude, based on the scalar potential of the edge flow field.

5.2.1 Edge Flow

In contrast, in our approach the detection and localization of edges (or image boundaries in a more general sense) are performed indirectly. First, by identifying a flow direction at each pixel location that points to the closest boundary, followed by the detection of locations that encounter two opposite directions of the edge flow. Since any of the image attributes such as colour, texture, or their combination can be used to define the edge flow, this scheme provides a general framework for integrating different types of image information for boundary detection.

5.3 SYSTEM DESCRIPTION

In this section, the processing steps of the proposed enhanced edge flow vector approach are presented. The algorithm consists of six stages of image acquisition: Gaussian Kernel Smoothing, Gabor filtering and Smoothing, Edge Flow vector, Edge Detection, and post-processing. Figure 5.1 shows the process flow of the proposed human motion detection algorithm. Each of these stages will be described in detail. The proposed rule
base should solve the problem of achieving a robust tracking, using the sparse and erroneous data from the simple image processing algorithms.

Figure 5.1  Block diagram of the enhanced edge flow method

5.3.1  Image Acquisition and Frame separation

Image acquisition aims to obtain image frames from a stationary or moving camera, or multiple cameras. Usually, a frame grabber is used to sub sample a sequence of video images at a certain frame rate, before the actual processing begins. In certain video systems, the acquisition process may be susceptible to erratic changes in illumination, reflection, and noise.

5.3.2  Gaussian Kernel Smoothing

Kernel smoothing is a group of powerful smoothing algorithms that involve applying a function known as the kernel to each data point in the
time-series. Kernel Smoothing belongs to the class of weighted moving averages.

This means, that in practice, all the points in the time-series are weighted, using as weights the results of the computation of the kernel function. Every kernel function has several properties:

- All Kernel values are positive or zero
- The kernel functions are normally symmetric
- Kernel function values decrease to zero from a central(maximum) value

Gaussian smoothing is often applied because the noise or the nature of the object observed might be of a Gaussian probable form. In general, image noise should be eliminated through image preprocessing. Then, some techniques such as the Gaussian filter are used to smooth the image to diminish the influence of noise.

5.3.3 Gabor Filtering and Smoothing

The Gabor filter output is used to construct an edge flow field for detecting boundaries.

This approach accurately detects the illusory boundaries. The complex Gabor filtered image can be written as

\[ O(x,y) = \text{Re}(x,y) + \text{Im}(x,y) \quad (5.1) \]

Where \( \text{Re}(x,y) \) and \( \text{Im}(x,y) \) represent the real and imaginary parts of the Gabor filtered output, respectively.
The phase of the filtered image can be expressed as
\[ \phi(x,y) = \text{atan} \left( \frac{\text{Im}(x,y)}{\text{Re}(x,y)} \right) \] (5.2)

The Gabor filter is basically a Gaussian (with variances \( s_x \) and \( s_y \) along x and y-axes respectively) modulated by a complex sinusoid (with centre frequencies \( U \) and \( V \) along x and y-axes respectively) and described by the following equation:

\[
G(x,y) = \frac{1}{2\pi s_x s_y} \cdot \exp \left( \frac{-1}{s_x^2 + s_y^2} \cdot x^2 + y^2 \right) \cdot \exp \left( 2\pi i (Ux +Vy) \right) \] (5.3)

**Description**

- **I**: Input image
- **Sx & Sy**: Variances along x and y-axes respectively
- **U & V**: Centre frequencies along the x and y-axes respectively
- **G**: is the output filtered image

### 5.3.4 Edge Flow method

(i) **Definition of the Edge Flow**

Let us define the general form of the edge flow vector at image location with an orientation as

\[ F(s,\theta) = F[E(s,\theta),P(s,\theta),P(s,\theta+\pi)] \] (5.4)

Where

- **E(s,\theta)** is the edge energy at the location along with the orientation.
P(s,θ) represents the probability of finding the image boundary if the corresponding flow at location s flows in the direction θ.

P(s,θ+π) represents the probability of finding the image boundary if the corresponding flow at location s flows backwards, i.e., in the direction θ+π.

The first component E(s, θ) of the edge flow is used to measure the energy of the local image information change (such as intensity/colour, texture, and phase difference), and the remaining two components P(s, θ) and are P(s+θ+π) used to represent the probability of the flow direction. The basic steps for detecting the image boundaries are summarized as follows:

- At each image location, first compute its local edge energy and estimate the corresponding flow direction.
- The local edge energy is iteratively propagated to its neighbour if the edge flow of the corresponding neighbour points in the same direction.
- The edge energy stops propagating to its neighbor, if the corresponding neighbour has an opposite direction of edge flow. In this case, these two image locations have both their edge flows pointing at each other, indicating the presence of a boundary between the two pixels.
- After the flow propagation reaches a stable state, all the local edge energies will be accumulated at the nearest image boundaries. The boundary energy is then defined as the sum of the flow energies from either side of the boundary.
(ii) Edge Flow Vector

The edge energies and the corresponding probabilities obtained from different image attributes can be combined together to form a single edge flow field for boundary detection.

Consider

\[ E(s, \theta) = \sum_{a \in A} E_a(s, \theta) \cdot w(a) \text{ and } \sum_{a \in A} W(a) = 1 \]  \hspace{1cm} (5.5)  

\[ P(s, \theta) = \sum_{a \in A} P_a(s, \theta) \cdot w(a) \]  \hspace{1cm} (5.6)  

where \( E_a(s, \theta) \) and \( P_a(s, \theta) \) represent the energy and probability of the edge flow computed from the image attribute \( a \in \{ \text{intensity/colour, texture, phase} \} \). \( w(a) \) is the weighting coefficient associated with the image attribute.

Now consider the use of the combined colour and texture information for boundary detection. For a given color image, the intensity edge flow can be computed in each of three color (RGB) bands. The flow direction needs to be estimated as well. At each location in the image, \( \{ [E(s, \theta), P(s, \theta), P(s, \theta + \pi)] \}_{\theta = 0}^{\theta = 2\pi} \) will be there. A continuous range of flow directions should be identified, which maximizes the sum of probabilities in the corresponding half plane:

\[ \Theta(s) = \arg \max_{\theta} \left\{ \sum_{\theta < \theta < \pi} \cdot P\left(s, \theta + \pi \right) \right\} \]  \hspace{1cm} (5.7)  

The edge flow vector is then defined to be the following vector sum:

\[ \vec{E}(s) = \sum_{\theta \in \Theta(s)} E(S, \theta) \cdot \exp(j^{\theta}) \]  \hspace{1cm} (5.8)
Where $\tilde{F}(s)$ is a complex number with its magnitude representing the resulting edge energy, and the angle representing the flow direction.

The main properties of the edge flow vector field can be summarized as follows:

- The vectors point normally towards the nearest edge;
- The magnitude of the vectors is small away from the edges and increases near the edges.
- The flow vectors from opposite directions cancel each other on the edges.

The flow direction at each pixel location that points to the closest boundary should be identified, followed by the detection of locations that encounter two opposite directions of the edge flow. Since any of the image attributes such as color, texture, or their combination can be used to compute the edge energy and direction of flow, this scheme provides a general framework for integrating different image features for boundary detection.

The Edge Flow method utilizes a predictive coding model to identify and integrate the direction of change in the image attributes, such as colour, texture, and phase discontinuities, at each image location.

5.3.5 Edge detection

Edge detection is a fundamental tool in image processing and computer vision, particularly in the areas of feature detection and feature extraction, which aim at identifying points in a digital image at which the image brightness changes sharply or, more formally, has discontinuities. The purpose is to recognize objects in a complex scene. Edge detection algorithms usually detect sharp transitions of intensity and/or colour within an image.
These transitions are characteristic of object edges. Once the edges of an object are detected, other processes such as region segmentation and object recognition can take place.

Much of the research on edge detection has been devoted to the development of optimal edge detectors, which provide the best trade-off between the detection and localization performance. A common strategy in designing such edge operators is to find the filter which optimizes the performance with respect to the three criteria: good detection, good localization, and a unique response to a single edge. In existing algorithms (Canny 1994) the optimal detector can be approximated by the first derivative of a Gaussian. By convolving the image with this filter, the edge detection is equivalent to finding the maxima in the gradient magnitude of a Gaussian-smoothed image in the appropriate direction. Detecting and combining edges at multiple resolutions and scales, is another important issue in edge detection. The scale-space technique introduced by Witkin (1983) involves generating coarser resolution images, by convolving the original images with a Gaussian smoothing kernel.

5.3.6 Post-processing

After boundary detection, disjoint boundaries are connected to form closed contours and result in a number of image regions.

After the edge flow of an image is computed, boundary detection can be performed by iteratively propagating the edge flow, and identifying the locations where two opposite direction of flows encounter each other. At each location, the local edge flow is transmitted to its neighbour in the direction of the flow, if the neighbour also has a similar flow direction. The steps are
1. Set \( n = 0 \) and \( \mathbf{F}_0(s) = \mathbf{F}(s) \)

2. Set the initial Edge flow \( \mathbf{F}_{n-1}(s) \) at time \( n+1 \) to zero

3. At each image location, identify the neighbor \( s' = (x', y') \) which is in the direction of the edge flow \( \mathbf{F}_n(s) \)

4. Propagate the edge flow if \( \mathbf{F}_n(s').F_n(s) > 0 \):
   \[
   \mathbf{F}_{n-1}(s') = \mathbf{F}_{n-1}(s') - \mathbf{F}_n(s)
   \]
   otherwise the edge flow stays at its original location \( \mathbf{F}_{n-1}(s) = \mathbf{F}_{n-1}(s) + \mathbf{F}_n(s) \)

5. If nothing has been changed, stop the iteration. Otherwise, set \( n = n+1 \) and go to step 2 and repeat the process.

Once the edge flow propagation reaches a stable state, detect the image boundaries by identifying the locations which have non-zero edge flows coming from two opposing directions. Let the edge signals \( V(xy) \) and \( H(xy) \) be the vertical and horizontal edge maps between the image pixels, and let

\[
\mathbf{F} = h((x, y), v(x, y)) = (\text{re}(\mathbf{F}(s)), \text{Im}(\mathbf{F}(s)))
\] (5.9)

**Figure 5.2** Edges and image pixels. (a) Image pixel \( I(x; y) \) is surrounded by two horizontal edges \( (H(x; y) \) and \( H(x; y +1) \) and two vertical edges \( V(x; y) \) and \( V(x +1; y) \). (b) The stable flow field vector \( \mathbf{F} \) and its projections \( h(x; y) \) and \( v(x; y) \) on the horizontal and vertical axes, respectively. (c) Boundary
detection based on the edge flow. The shaded rectangles indicate the edges which lie between the image pixels.

Then, the edge signals \( V(x,y) \) and \( H(x,y) \) will be turned on, if and only if, the two neighboring edge flows point at each other. Once the edge signal is on, its energy is defined to be the summation of the projections of those two edge flows towards it. Summarizing:

- Turn on the edge \( V(x,y) \) if and only if \( H(x-1,y) \geq 0 \) and \( H(x,y) < 0 \); then \( V(x,y) = H(x-1,y) - H(x,y) \) \hspace{1cm} (5.10)

- Turn on the edge \( H(x,y) \) if and only if \( V(x,y-1) \geq 0 \) and \( V(x,y) < 0 \); then \( H(x,y) = V(x,y-1) - V(x,y) \) \hspace{1cm} (5.11)

After the edge signals are detected, the connected edges are used to form a boundary, whose energy is defined to be the average of its edge signals \( V(x,y) \) and \( H(x,y) \) are shown in Figure 5.2.

After boundary detection, disjoint boundaries are connected to form closed contours, and result in a number of image regions. The basic strategy for connecting the boundaries are summarized as follows.

- For each open contour, associate a neighbourhood search size proportional to the length of the contour. This neighbourhood is defined as a half circle with its center located at the unconnected end of the contour.

- The nearest boundary element which is within the half circle is identified.
If such a boundary element is found, a smooth boundary segment is generated to connect the open contour to the nearest boundary element.

This process is repeated a few times (typically 2-3 times), till all the salient open contours are closed.

At the end, a region merging algorithm is used to merge similar regions, based on a measurement that evaluates the distances of region color and texture features, the sizes of the regions, and the percentage of the original boundary between the two neighbouring regions. This algorithm sequentially reduces the total number of regions each time, by checking if the user’s preferred number has been approached to the best extent.

5.4 EXPERIMENTS AND ANALYSIS

5.4.1 Data acquisition and ground-truth

In the experiments three videos have been chosen from the PETS dataset, depicting an area near an airport terminal building. Video 1- target walking in the shaded region, Video 2- target walking from the shaded region towards the bright sunny region, and Video 3- target walking from the sunny bright region towards the shaded region. Ground truth is taken from the website at http://impca.cs.curtin.edu.au/downloads.php to test the performance of the proposed system in terms of VSA.

5.4.2 Performance Analysis

An ideal edge segmentation would produce a set of connected edge points, that match exactly those edge points in the target given by the ground truth. An effective way of determining the quality of the information provided by a feature can be summarized by a frame by frame precise analysis of the
edges at the pixel level using measures: Video Sequence Accuracy (VSA) defined as:

\[ VSA = \frac{1}{N} \sum_{i=0}^{n} \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i} \] (5.12)

Where TP (true +ve) is a ground truth edge pixel of the object inside the bounding box correctly detected as an object pixel in the image, TN(true -ve) is a ground truth background pixel inside the bounding box, correctly detected as a background pixel in the image, FN (false -ve) is a ground truth object pixel of the object inside the bounding box, wrongly detected as a background pixel in the test frame, and FP (false +ve) is a ground truth background pixel inside the bounding box, wrongly detected as an object pixel in the test frame. The VSA averages the accuracy and precision, computed per frame, over the sequence of \( N \) images.

There are two processes used to get the ground truth. The first is to determine the bounding box around the target of interest (a person). This simulates the process of predicting the region of interest in which the person would appear in the next image in the sequence. The second is to determine the edges inside the bounding box that describe the target. The bounding box that encloses the target is defined manually. This process is quicker than simply manually defining the edge of the target. A pixel by pixel comparison in the region defined by the BB was carried out on a frame to frame basis, and the edge pixels were classified to produce TPi, TNi, FNi and FPi.

The proposed methods are demonstrated and compared with recently published (Gladis et al 2009) five edge detectors (Marr-Hildreth, Canny, Sobel, Roberts and Prewitt) under varying lighting conditions.

5.4.3 Results
Figures 5.3-5.5 a) shows the input image. Figures 5.3-5.5 b) shows the corresponding ground truth image human was tracked correctly and compared with ground truth. Figures 5.3-5.5 c) shows the detected image for input image. Figures 5.3-5.5 d) shows the corresponding human was tracked correctly with original bounding box. Figures 5.3-5.5 e) shows the corresponding human was tracked correctly with the detected bounding box. Table 5.1 and Figure 5.6 shows the VSA measures of the proposed method, compared with the existing edge detection methods.

Figure 5.3 Segmentation Result of Video 1: (a) Initial frame of sequence (b) ground truth (c) resulting edges obtained from the proposed method (d) original bounding box image (e) resulting detected bounding box obtained from the proposed method.

Figure 5.4 Segmentation Result of Video 2: (a) Initial frame of sequence (b) ground truth (c) resulting edges obtained from the proposed method (d) original bounding box image (e) resulting detected bounding box obtained from the proposed method.
Figure 5.5  Segmentation Result of Video 3 (a) Initial frame of sequence (b) ground truth (c) resulting edges obtained from the proposed method (d) original bounding box image (e) resulting detected bounding box obtained from the proposed method.

Table 5.1 Comparison of VSA values of the edge flow vector and other methods

<table>
<thead>
<tr>
<th>Video</th>
<th>Video Sequence Accuracy(VSA)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Canny</td>
</tr>
<tr>
<td>1</td>
<td>0.912</td>
</tr>
<tr>
<td>2</td>
<td>0.926</td>
</tr>
<tr>
<td>3</td>
<td>0.957</td>
</tr>
</tbody>
</table>

Figure 5.6  Graph for VSA values of Target 1, 2 and 3 frame accuracy: Canny (magenta), Prewitt (black), Roberts (blue), Sobel (yellow), Marr-Hildreth (green) & Proposed (red).
The novel framework for detecting image boundaries demonstrated its use in segmenting a large variety of natural images. In contrast to the traditional approaches, the edge flow model utilizes a predictive coding scheme to detect the direction of change in various image attributes and construct an edge flow field. By iteratively propagating the edge flow, the boundaries can be detected at image locations, which encounter two opposite directions of the flow in the stable state. The only significant control parameter is the image scale, which can be adjusted to the user’s requirements. For simplicity, a single scale parameter has been used for the entire image segmentation in our current implementation. However, this might not be appropriate for some images which contain multiple scale information. There is a need to locally adjust the scale parameter depending on the properties, such that meaningful boundaries at each image location can be detected.

However, the experimental results give better than most of the algorithms that are currently available on segmentation literature. A visual inspection of the results indicates that the segmentation is of acceptable quality. The accuracy of the proposed edge detection method is high for all the existing edge detectors over the three sequences, showing a good classification of edge and non-edge pixels. The precision is poor for all edge detectors because of many spurious edge pixels detected. Marr-Hildreth (Gladis John et.al 2009) is the worst, and along with Canny, is outperformed by Prewitt and Sobel. The results are generally similar (except for Marr-Hildreth) and there is a little fluctuation when the target is either in shadow or in sunlight. Better results occur when the target is in sunlight (increasing for video 2 and decreasing for video 3). In video 1, there is a sharp drop in precision near the beginning, because the target is carrying a bag that is of the same colour as the background at that region in the scene, resulting in a reduction in the detected target edge pixels.
5.5 SUMMARY

The new improved edge flow vector based segmentation for people tracking provides detailed experimental results, that demonstrated the advantage of the edge flow vector over other methods. There are a number of measures to access the segmentation performance of low level features in the context of model based recognition and tracking of people. There is some variation in the results for the edge detectors, when considering changes in illumination. During the experiments, it was observed that

- The improved edge flow vector works for different kind of videos.
- The improved edge flow vector method produces a better segmented object

It was also observed that the edge flow method does not show encouraging results for the following variations.

- The bounding box is pre-defined for the video.
- The edge flow method comparison cannot work without the ground truth.

Consequently, all these observations, drawbacks and advantages were analyzed. The next chapter focuses on the effective object tracking method. It describes the robust and stable tracking framework for counting people. It also describes the integration of the background subtraction, segmentation and tracking methods.