ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/306357420

Comparison of Some Motion Detection Methods in cases of Single and Multiple Moving Objects

Article · October 2012

CITATIONS
6

READS
2,329

1 author:

Shamir Alavi
Canada Revenue Agency
3 PUBLICATIONS **32** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Quaternion-Based Gesture Recognition Using Wireless Wearable Motion Capture Sensors View project

All content following this page was uploaded by Shamir Alavi on 22 August 2016.

The user has requested enhancement of the downloaded file.

Comparison of Some Motion Detection Methods in cases of Single and Multiple Moving Objects

Shamir Alavi *alavi1223@hotmail.com*
Electrical Engineering
National Institute of Technology Silchar
Silchar 788010 (Assam), India

Abstract

Motion detection tells us whether there is a change in position of an object with respect to its surroundings or vice versa. It is applied to various domestic and commercial applications starting from simple motion detectors to high speed video surveillance systems. In this paper, results obtained from some simple motion detection algorithms, which use methods like image subtraction and edge detection, have been compared. The software used for this purpose was MATLAB 7.6.0 (R2008a). It has been observed that while image subtraction is sufficient to detect motion in a video stream, combining it with edge detection in different sequences yields different results in different scenarios.

Keywords: Motion Detection, Edge Detection, Canny, Sobel, Image Subtraction, Computer Vision

1. INTRODUCTION

Motion detection is a process of confirming a change in position of an object relative to its surroundings or the change in the surroundings relative to an object [1]. It has paramount importance in any vision based detection and tracking system. Throughout the last couple of decades, several techniques have been introduced to accomplish this task effectively. However, there is no perfect system or method which can overcome the various problems that are faced during detection. The difficulties are generally associated with lighting condition of the surrounding, illumination of the object itself which is to be detected, speed of its movement or the type of object [2].

Generally, motion detection is useful in real time or active video surveillance systems [2]. In this paper, the main focus is given to the processing of the captured video data to detect motion in it. Two main methods, image subtraction and edge detection have been used for detection. Three different cases have been considered in order to compare the results. One detects the motion by image subtraction only whereas the other two include edge detection in different sequence. Moreover, two different edge detection techniques, namely Sobel edge detection [3] and Canny edge detection [4] have also been taken into account while comparing the results. Finally, all of these are implemented on two different scenarios where the number of moving object is one or more. After a brief review of image subtraction and the two edge detection techniques, the motion detection algorithms used here along with comparisons in regard to the scenarios and edge detection techniques will follow accordingly.

2. Image Subtraction

Image subtraction is one of the popular techniques in image processing and computer vision. Basically image subtraction can be represented as:

$$\Delta I(i,j) = I_{Curr}(i,j) - I_{Prev}(i,j) \quad (1)$$

where:

$\Delta I(i,j)$ is the difference in image intensity between two consecutive frames. $I_{Curr}(i,j)$ and $I_{Prev}(i,j)$ represent image intensities for current and previous frames respectively [2].

(a) (b) (c)

FIGURE 1: (a) Frame 1 (b) Frame 2 (c) Image after subtraction

This is primarily done for one of two reasons – leveling uneven sections of an image such as half an image having a shadow on it, or detecting changes between two images. This detection of changes can be used to tell if something in the image moved [5].

3. Edge Detection

Edge detection refers to the process of identifying and locating sharp discontinuities in an image. The discontinuities are abrupt changes in pixel intensity which characterize boundaries of objects in a scene [6]. The purpose of edge detection in general is to significantly reduce the amount of data in an image, while preserving the structural properties to be used for further image processing [7]. Classical methods of edge detection involve convolving the image with an operator (a 2-D filter), which is constructed to be sensitive to large gradients in the image while returning values of zero in uniform regions [6]. There are various ways to perform edge detection as various techniques have been introduced throughout the years. This work will compare two such techniques while detecting motion which are the Sobel operator or Sobel edge detection [3] and the Canny edge detection [4]. There are several other operators also such as Prewitt's operator, Robert's cross operator, Laplacian of Gaussian etc. but most of them (such as Prewitt's and Roberts's operator) work in a fashion similar to Sobel operator [6].

3.1 Sobel Operator

The operator consists of a pair of 3×3 convolution kernels as shown in Figure 1. One kernel is simply the other rotated by 90°.

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

G_x

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

G_y

FIGURE 1: Masks used by Sobel Operator

These kernels are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (let us call these G_x and G_y). These can then be combined to find the absolute magnitude of the gradient at each point and the orientation of that gradient [3]. The gradient magnitude is given by:

$$|G| = \sqrt{(G_x^2 + G_y^2)} \quad (2)$$

Typically, an approximate magnitude is computed using:

$$|G| = |G_x| + |G_y| \quad (3)$$

which is much faster to compute.

The angle of orientation of the edge (relative to the pixel grid) giving rise to the spatial gradient is given by:

$$\theta = arctan(G_y / G_x) \quad (4)$$

3.2 Canny Edge Detection

The algorithm runs in five separate steps [7]:

i. **Smoothing**: Blurring the image to remove noise.
ii. **Finding gradients**: The edges are marked where the gradients of the image has large magnitudes.
iii. **Non-maximum suppression**: Only local maxima are marked as edges.
iv. **Double thresholding**: Potential edges are determined by thresholding strong and weak edges.
v. **Edge tracking by hysteresis**: Final edges are determined by suppressing all the edges that are not connected to a very certain (strong) edge.

(a) (b) (c)

FIGURE 2: (a) Original image (b) Sobel edge (c) Canny edge

4. Algorithms Used for Motion Detection

Using image subtraction and edge detection as the main tool, following three algorithms have been used:

4.1 Image Subtraction Method

The first task is to extract frames from the continuous video stream so that they can be processed further for our next tasks. The steps for this algorithm are stated below:

i. Extract frames from video stream.
ii. Write the extracted frames as image files.
iii. Subtract the previous image from current image as stated in (1).
iv. Convert image to binary image.
v. Label connected components.
vi. Perform blob analysis (i.e. measure properties of each labeled image regions).
vii. Calculate the centre of mass of each labeled region and label it (to detect as many moving elements as possible).
viii. Play the labeled images as a continuous video stream to detect motion.

4.2 Edge Detection after Image Subtraction

In this algorithm, edge detection is performed after image subtraction. The algorithm is as follows:

i. Extract frames from video stream.
ii. Write the extracted frames as image files.
iii. Subtract the previous image from current image.
iv. Convert the image to grayscale.
v. Detect edges.
vi. Label connected components.
vii. Perform blob analysis.
viii. Calculate the centre of mass of each labeled region and label it.
ix. Play the labeled images as a continuous video stream to detect motion.

4.3 Image Subtraction after Edge Detection

In this case, edge detection is performed before image subtraction. The algorithm is stated below:

i. Extract frames from video stream.
ii. Write the extracted frames as image files.
iii. Detect edges in all the images.
iv. Subtract the previous image from current image.
v. Convert subtracted image to binary image.
vi. Label connected components.
vii. Perform blob analysis.
viii. Calculate the centre of mass of each labeled region and label it.
ix. Play the labeled images as a continuous video stream to detect motion.

5. Comparison of the Motion Detection Algorithms

5.1 Considered Scenarios

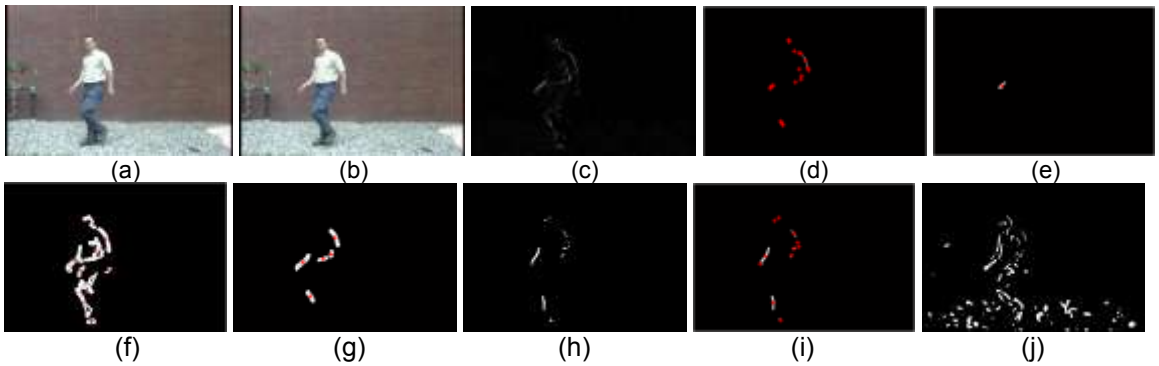
Two different scenarios have been considered for comparison:

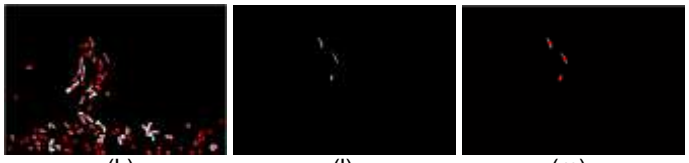
Scenario 1: A man is walking while everything else is still. Therefore, we have a single moving object in this scenario (as in Figure 1)

Scenario 2: Plastic caps are being collected from one conveyor belt to another. Here, the caps as well as the belts are in motion. Hence, we have multiple moving objects in this scenario (as in Figure 2).

5.2 Visual Comparison

From the extracted frames, two consecutive frames have been taken for comparison and analysis:

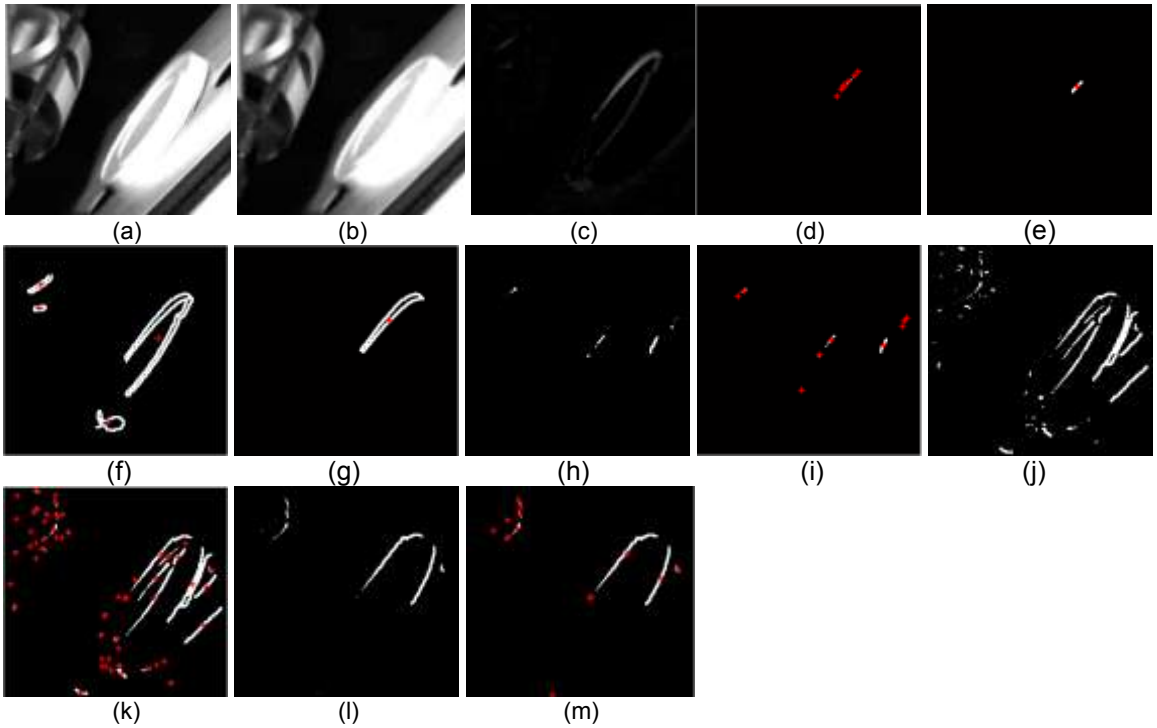




(k) (l) (m)

FIGURE 3: Comparison of motion detection algorithms on scenario 1, (a) and (b) two consecutive frames (c) subtracted image (d) algo* 4.1 (e) algo 4.2 – Sobel (thresh = 0.17) (f) algo 4.2 – Canny (thresh = 0.17) (g) algo 4.2 – Canny (thresh = 0.55) (h) subtracted image of Sobel edges (i) algo 4.3 – Sobel (thresh = 0.17) (j) subtracted image of Canny edges (thresh = 0.17) (k) algo 4.3 – Canny (thresh = 0.17) (l) subtracted image of Canny edges (thresh = 0.55) (m) algo 4.3 – Canny (thresh = 0.55)

*algo = algorithm



(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

(k) (l) (m)

FIGURE 4: Comparison of motion detection algorithms on scenario 2, (a) and (b) two consecutive frames (c) subtracted image (d) algo 4.1 (e) algo 4.2 – Sobel (thresh = 0.17) (f) algo 4.2 – Canny (thresh = 0.17) (g) algo 4.2 – Canny (thresh = 0.55) (h) subtracted image of Sobel edges (i) algo 4.3 – Sobel (thresh = 0.17) (j) subtracted image of Canny edges (thresh = 0.17) (k) algo 4.3 – Canny (thresh = 0.17) (l) subtracted image of Canny edges (thresh = 0.55) (m) algo 4.3 – Canny (thresh = 0.55)

N.B.: For Canny edges, higher threshold = thresh, lower threshold = 0.4 * thresh, standard deviation of the Gaussian filter = 1. For instance, in figure 4(j), higher threshold = 0.17, lower threshold = 0.4 * 0.17 = 0.068, standard deviation of the Gaussian filter = 1.

5.3 Comparison Tables

The following table summarizes the visual comparisons shown above:

Algorithm	Edge	Scenario 1	Scenario 2
4.1		Detects motion quite well despite data loss due to binary conversion of the subtracted image - fig 3(d)	Partially detects motion of only one cap - fig 4(d).

4.2	Sobel	Minor visual detection – fig 3(e).	Minor visual detection – fig 4(e).
	Canny (thresh=0.17)	Great visual detection (detects movement of the whole body) – fig 3(f).	Unable to detect the motion of the belts (but better than Sobel as in fig 4(e)) – fig 4(f).
	Canny (thresh=0.55)	Detection of strong edges only (better than Sobel as in fig 3(e)) – fig 3(g).	Unable to detect the upper cap and the belts (still better than Sobel as in fig 4(e)) – fig 4(g).
4.3	Sobel	Detection quality almost similar to that of algo 4.1 -fig 3(i).	Partially detects both the caps but cannot detect movements of the belts – fig 4(i).
	Canny (thresh=0.17)	Detects movement of the man but erroneously detects several portions of the still pavement as moving objects – fig 3(k).	Great visual detection – detects movement of both the caps and the lower belt; slightly detects movement of the upper belt as well – fig 4(k).
	Canny (thresh=0.55)	Detection of strong edges only (no false detection) – fig 3(m). However, detection quality is lower than that of Sobel detection as in fig 3(i).	Unable to detect the motion of the belts – fig 4(m). However, detection quality is better than that of Sobel detection as in fig 4(i).

TABLE 1: Comparison of the motion detection algorithms under different scenarios

Average** time taken to obtain the above shown figures (only the ones in which motions are detected):

Algorithm	Edge	Scenario1***		Scenario 2****	
		Figure	Average time(sec)	Figure	Average time(sec)
4.1		3 (d)	3.0717644	4 (d)	4.1041660
4.2	Sobel	3 (e)	3.7338478	4 (e)	6.4473798
	Canny (thresh = 0.17)	3 (f)	4.0066358	4 (f)	6.4270532
	Canny (thresh = 0.55)	3 (g)	3.8619630	4 (g)	6.3946814
4.3	Sobel	3 (i)	7.5335376	4 (i)	13.8421282
	Canny (thresh = 0.17)	3 (k)	13.3262350	4 (k)	26.4577750
	Canny (thresh = 0.55)	3 (m)	12.6463506	4 (m)	25.0108116

TABLE 2: Comparison of average time taken by the algorithms to detect motion

** Each program has been run 5 times on the same system (Intel® Core(TM) i5 CPU M430 @ 2.27 GHz, 4GB DDR3 RAM, ATI Radeon HD 5400, Win 7 64-bit) and their average was considered.
*** Total number of frames in the video stream for this scenario = 80
**** Total number of frames in the video stream for this scenario = 249

6. Result and Discussion

From the comparisons shown above in table 1, it is clear that image subtraction only is good enough to detect motion in case of a single moving object in front of a still background. However, it fails to accomplish this task properly when there are multiple moving objects.

For a single moving object, the best result has been obtained by performing Canny edge detection after image subtraction where a low threshold value was used. Sobel edge detection could not perform as well as Canny edge detection. On the other hand, in case of multiple moving objects, the best result came from performing Canny edge detection before image subtraction. Here also, a low threshold value was used.

From table 2, we can see that 'Image Subtraction Method' is the fastest of the three motion detection methods that are compared here. Between the other two methods, performing edge detection after image subtraction is substantially faster. A careful look at the average times taken by this method also tells us that it is not that slow in comparison to the simpler image subtraction method (extra time taken by algo 4.2 as compared to 4.1 is below 1 second for scenario 1 and just above 2 seconds for scenario 2). For multiple moving objects, though Canny gave the best result, it came at the cost of high computation time (26.4577750 seconds).

7. Conclusion and Future Research

To sum up, edge detection, in addition to image subtraction is necessary to detect motion properly. Canny edge detection, in spite of being computationally slower and more expensive as a result, gives the best outcome under any circumstances. On top of it, 'Image Subtraction after Edge Detection' is the best method out of the three discussed above if we can compromise the higher computational time taken by it.

For further research, only Canny edge detection will be used as it has been deemed to work best among all other edge detection methods available at present [8, 9] and is referred to as a 'modern standard' [10]. It is also necessary to improve the higher computation times with Canny edge detection, especially in complex situations with multiple moving objects. This research work can be further extended by testing under different lighting conditions, differing the distance of the moving objects and finally going live with a faster algorithm and system.

8. References

[1] Wikipedia. "Motion Detection." Internet: http://en.wikipedia.org/wiki/Motion_detection, May 20, 2012 [Jun. 16, 2012].

[2] R. B. Wahyu, Tati R. Mengko, Bambang Pharmasetiawan and Andryan B. Suksmono. "Motion Detection Using Image Subtraction Edges Detection." Risalah Lokakarya Komputasi dalam Sains dan Teknologi Nuklir XVII, pp. 157-171, Aug. 2006.

[3] J. Matthews. "An Introduction to Edge Detection: The Sobel Edge Detector." Internet: http://www.generation5.org/content/2002/im01.asp, 2002.

[4] J. Canny. "A Computational Approach to Edge Detection." *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679-697, Nov. 1986

[5] Wikipedia. "Image subtraction." Internet: http://en.wikipedia.org/wiki/Image_subtraction, Sep. 27, 2011 [Jun. 16, 2012].

[6] R. Maini and Dr. H. Aggarwal. "Study and Comparison of Various Image Edge Detection Techniques." International Journal of Image Processing (IJIP), vol. 3, iss. 1, pp. 1-11, Feb. 28, 2009.

[7] "Canny Edge Detection." Internet: http://www.cvmt.dk/education/teaching/f09/VGIS8/AIP/canny_09gr820.pdf, Mar. 23, 2009.

[8] E. Nadernejad, S. Sharifzadeh and H. Hassanpour. "Edge Detection Techniques: Evaluations and Comparisons." Applied Mathematical Sciences, vol. 2, no. 31, pp. 1507 – 1520, 2008.

[9] M. Juneja and P. S. Sandhu. "Performance Evaluation of Edge Detection Techniques for Images in Spatial Domain." International Journal of Computer Theory and Engineering, vol. 1, no. 5, pp. 614-621, Dec. 2009.

[10] M. Heath, S. Sarkar, T. Sanocki and K. Bowyer. "Comparison of Edge Detectors: A Methodology and Initial Study." Computer Vision And Image Understanding, vol. 69, no. 1, pp. 38–54, Jan. 1998.

View publication stats