International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:9, No:9, 2015

A Real-Time Image Change Detection System

Madina Hamiane, Amina Khunji

Abstract**—**Detecting changes in multiple images of the same scene has recently seen increased interest due to the many contemporary applications including smart security systems, smart homes, remote sensing, surveillance, medical diagnosis, weather forecasting, speed and distance measurement, post-disaster forensics and much more. These applications differ in the scale, nature, and speed of change. This paper presents an application of image processing techniques to implement a real-time change detection system. Change is identified by comparing the RGB representation of two consecutive frames captured in real-time. The detection threshold can be controlled to account for various luminance levels. The comparison result is passed through a filter before decision making to reduce false positives, especially at lower luminance conditions. The system is implemented with a MATLAB Graphical User interface with several controls to manage its operation and performance.

Keywords**—**Image change detection, Image processing, image filtering, thresholding, B/W quantization.

I. Introduction

DETECTING changes in image frames is part of many contemporary applications. These include smart security systems, smart homes, weather forecasting, automatic people counting, object detection, object tracking, speed and distance measurement, action recognition, irregularity detection, post-disaster forensics and much more [1]. Many of today's digital cameras have some form of change detection to either change focus or shutter settings or to capture specific types of action. These applications differ in the scale, nature, and speed of change. They also differ in their cost; while some may tolerate false positives others may not. Some applications can employ controlled amount of luminance while others cannot, for instance, in outdoors applications.

The major challenge that faces image change detection is avoiding false positives without introducing false negatives False positives are defined as the number of no-change pixels incorrectly detected as change (also known as false alarms) while false negatives represent the number of change pixels incorrectly detected as no-change (also known as misses) [2]. The main sources of false changes are:

- Noise: This happens all the time due to changes in sensor sensitivity over time that which in turn is due to several factors such as temperature, and power stability. It also happens because of very small and fast fluctuations in object luminosity due to power and temperature changes and other factors. In most cases this type of noise appear as "the salt and pepper noise" which as discussed later is managed differently from uniform white noise.
- External Illumination Changes: this may change over time due to many factors either generally or locally on parts of the observed scene and contribute to false change detection. Clouds, light source fluctuation, external object shadows are some to mention. This, especially drastic illumination changes tend to be difficult to overcome.
- Very low Algorithm threshold: The algorithm that would detect the change will have as one crucial parameter the threshold at which detection is considered [2]. If set too low, it would raise the false positives probability. If set too high it would increase "false negatives". Thus, the threshold needs to be set carefully and tuned to suit the application scenario [3].

Ideally, change detection is a simple process involving two image frames and a change is identified if the two frames are not exactly equal. However practical change identification mandates some pre-processing and thresholding to differentiate true changes from false ones resulting from uncontrolled factors [2]-[4]. Additionally, if the change rate is high, high performance real time computing while using efficient algorithms is necessary.

Image change detection techniques have been widely discussed in the literature [4]-[7]. Among the various reported techniques, image differencing to be a simple, straightforward technique which allowed for an easy interpretation of results and which can be applied to detect a wide range of image changes ranging from environmental changes to moving objects. Although only the magnitudes of these changes can be obtained, the method has proven its robustness [6] and shown to give results of high accuracy [7].

The main aim of this work is to develop a simple real-time image change detection system using Matlab image processing capabilities. The system is designed to detect changes in a video stream captured in real time from a camera. Image differencing technique is adopted for its simplicity, accuracy, robustness and wide scope of application. The proposed system aims to reduce false positives, account for low luminance situations, and to provide a user interface with real time configurable controls to enable fine tuning.

II. Change Detection Algorithm

The proposed change detection algorithm is illustrated in Fig. 1. Change is identified by comparing the RGB representation of two consecutive frames captured in real-time from a camera. The detection threshold is controlled to account for various luminance levels. The comparison result can be passed through a filter before decision making to reduce false positives, especially at lower luminance conditions. Once a change is successfully detected, a warning

Madina Hamiane is with the department of Telecommunication Engineering, Ahlia University, Bahrain (e-mail: mhamiane@ahlia.edu.bh).
Amina Khunji was an undergraduate student in the department of Computer Engineering, Ahlia University, Bahrain (e-mail: ameena-210@hotmail.com).

sign is issued both in visual and audio format. The detected changes are then saved as video files on the computer storage.

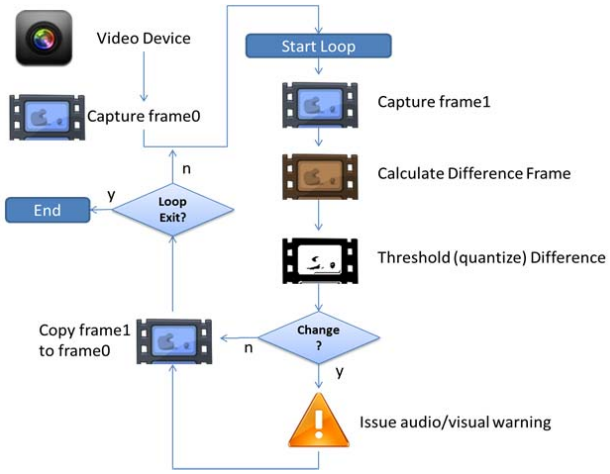

Fig. 1 Change detection Algorithm

All different phases of the algorithm were implemented using MATLAB Image Acquisition and Image Processing toolboxes. The algorithm is described in detail in the following steps:

Capture Initial Frame: In every loop, two consecutive frames need to be compared; the last frame from the previous iteration (frame0) and a new frame captured in the current iteration (frame1).

Calculate Difference: The built-in MATLAB function `imabsdiff` is used here. The function takes two color frames and returns the difference color frame:

```
Diff_Img=imabsdiff(frame0,frame1);
```

For instance, Fig. 2 shows two frames and the result of applying `imabsdiff`.

Fig. 2 `imabsdiff` function result

Thresholding (B/W Quantization): Observing the results of several experiments under different lighting conditions and types of changes, we noticed that to rely on this difference we need to set a threshold. The threshold value would be different for different setup and lighting conditions and the best value to balance having false negatives and false positives is difficult to set. For this reason we have chosen to normalize the difference by quantizing the difference image to a black and white image. This choice implied using a threshold value between 0 and 1 and after several tests, 0.5 was found to be the most suitable threshold value for most scenarios. We used Matlab function `im2bw` on the difference image. The result is shown in Fig. 3.

Decision Making: Another advantage of Quantizing to black and white (B/W) is allowing decision making using 0. That is, any presence of white pixels is interpreted as change. This is unlike using the difference color image alone which needs the comparison to be done at a threshold that might have to be changed from scenario to scenario. In the case of B/W quantization, the thresholding is taken away from the decision making step to the processing stage.

To account for all pixels, all BW image pixels are averaged using the matrix mean operation on two dimensions (Width-Height). This presents another advantage of B/W related to performance as in the color case we need to average on three dimensions (width-height-RGB).

```
Diff_Img_mean=mean(mean(BW));

if Diff_Img_mean>0
```

Positive Action: Once a change is decided, an action needs to be taken such as an audible or visual warning. Both warnings have been used. For the audible warning a wav file is read from disk and played using MATLAB.

```
[y,Fs]=audioread('wav\chattell.wav');
sound(y,Fs);
```

Looping: After decision making and positive action, the loop goes to the next iteration, copying frame1 to frame0 to obtain new frame1 in the next iteration and to repeat the algorithm.

Fig. 3 `im2bw` function result

III. The System GUI

The general components of the system's GUI are shown in Fig. 4. In general our GUI has two types of controls: Configuration and On/Off Functionality controls.

Configuration controls: The controls are accessible to the user to adjust the program performance.

- Frame Rate: to set the frame rate at which frames are captured.
- Filter: to choose filtering action before BW quantization.
- Minimum Luminance Change Magnitude: This sets the BW threshold for white color, as explained before.

On/Off Functionality controls: These turn on/off specific functionalities, we have two of them:

- Turn on/off monitoring
- Turn on/off recording

The rest of the GUI components do not allow user interaction. These are:

Image Frames: There are three image frames, each displays in real time the image frames of

- the current captured frame
- the difference
- the Black and White quantized image

Positive Action Animation: This is a fourth image frame, but using the timer function will display three consecutive frames to animate a red exclamation mark ("!") during change detection. It displays a black frame otherwise.

The Timer Function: The timer function is mostly the heart of any GUI, as it does the continuous action intended. In our case, the timer function does the frame looping explained before.

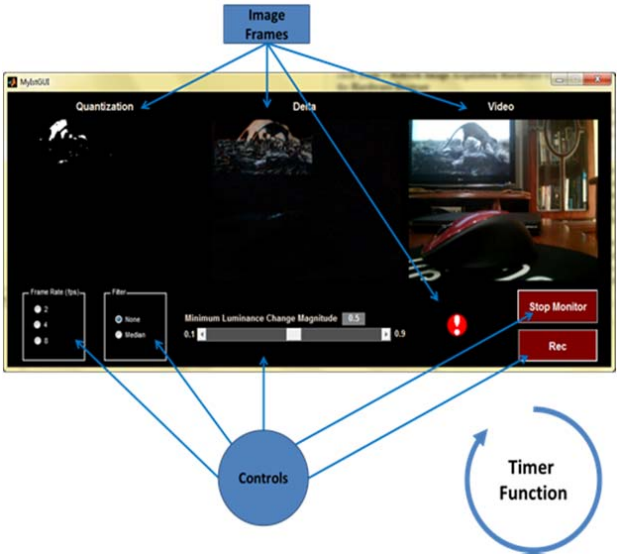

Fig. 4 System's GUI

IV. Optimization

To enhance false positives/negatives avoidance and to make the system more useful the following were applied.

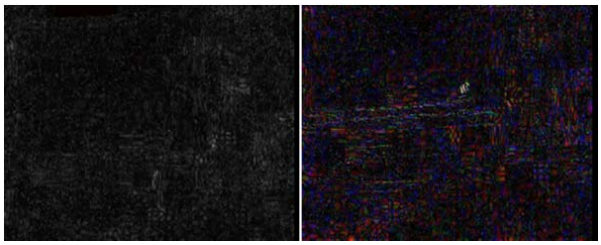

Fig. 5 Illustration of Median filter effect

A. Median Filtering Option

The difference image is passed into a Median filter to get rid of luminance noise before quantizing to black and white. This noise appears as "salt and pepper" noise and hence the use of the median filter. Luminance fluctuations are filtered out using this filter. Before the filter, the image is converted first to Grayscale.

Fig. 5 shows the presence of a median filter on the no change case. The image brightness is exaggerated 80% for illustration. On the right side, where no filter is applied we can observe the spot of luminance fluctuation close to the center. Such a fluctuation is filtered out using the median filter as shown on the left.

B. Video Recording Option

During changes, if this option is on, the continuous changing frames are saved to disk as a video. This would be useful in real life applications as it makes tracking the changes much easier than if the camera record was fully saved. This of course would also save disk and processing resources as the recording operation would practically be mostly idle.

V. System Testing and Results

The system was tested in various scenarios, but only indoors. The tests were performed using the following settings:

- Threshold: 0.18, 0.5, 0.82
- Luminance: low, medium, high indoor luminance
- Filter: none and median
- Change types: Person movement, smoke, TV movie

Table I shows the false negative rates for all the possible combinations of the above. The results were as expected :

- High threshold is almost useless all the time.
- Low threshold works fine but at the expense of higher risk of false positives
- Smoke requires low threshold and could be detected at low luminance.

A. High Luminance Situation

During day light or using high luminance, the precision in change detection is high. However, the false positive rate is also higher than the case of low luminance although still small. The presence of the median filter highly enhances this.

TABLE I
TESTING RESULTS

	Threshold	0.18			0.5			0.82		
	Luminance	low	med	high	low	med	high	low	med	high
No Filter	Change (person passing)	0%	0%	0%	80%	0%	0%	100%	70%	90%
	Change (smoke)	20%	20%	20%	80%	80%	20%	100%	100%	100%
	Change (TV movie)	5%	0%	0%	75%	0%	0%	100%	30%	100%
Median Filter	Change (person passing)	0%	0%	0%	90%	0%	10%	100%	90%	95%
	Change(smoke)	20%	20%	20%	90%	80%	40%	100%	100%	100%
	Change (TV movie)	5%	0%	0%	90%	5%	0%	100%	50%	100%

Observing one minute false positive detection at threshold 0.1 (very low threshold intentionally) resulted in 10 false changes/minute without the filter. With the filter, the rate dropped dramatically to almost zero.

B. Low Luminance Situation

In low luminance there seemed more risk of false negatives. To overcome that, the threshold was set to lower than 0.5.

VI. CONCLUSION

Current technology and computing power allow a great deal of usefulness in image processing. In this work a system was developed to detect changes in a video stream captured in real-time and was shown to effectively detect these changes in addition to significantly reduce false positives, account for low luminance situations, and provide fine tuning capabilities through a user interface with real time configurable controls.

It was shown and emphasized that the performance of the change detection algorithm could easily be enhanced with the use of:

- Filters
- Luminance control
- Threshold control

Depending on luminance/threshold setup there could be various applications. For instance, a controlled high luminance/low threshold would result in high sensitivity to dim changes such as in smoke detection.

The performance of the proposed system could be further enhanced with the use of a better image capture device quality and a higher computing power to allow complex algorithms to make more precise decisions.

The use of adaptive filters would further enhance the system's performance as these automatically adjust their performance depending on the luminance of the local part of the image and would allow outdoor application. It is also believed that automatically setting the threshold, or auto thresholding, depending on the luminance level may also enhance system performance. This would provide even further enhancement if performed together with adaptive filtering.

REFERENCES

[1] RI Capó Irizarry, "Fundamentals & Applications of Image Change Detection," *Research Thrust R2 Presentations. Northeastern University,* Paper 20, 2006.

[2] Richard J. Radke et al, "Image Change Detection Algorithms: A Systematic Survey," *IEEE Transactions on Image Processing,* Volume 14 , Issue 3, pp.294 – 307, 2005.

[3] P.L.Rosin, E.Ioannidis, "Evaluation of global image thresholding for change detection," *Pattern Recognition Lett.* Vol. 24, Issue 14, pp 2345–2356, 2003.

[4] T. A. Pawar, "Change Detection Approach for Images Using Image Fusion and C-means Clustering Algorithm," 4th *International Journal of Advance Research in Computer Science and Management Studies*, Volume 2, Issue 10, pp.303-307, 2014.

[5] M. Ilsever, C. Unsalan, "Pixel-Based Change Detection Methods, *Remote Sensing Aplications,*" Vol *X,* pp 7-21, 2012, Springer.

[6] Ashok Sundaresan, *et al*., "Robustness of Change Detection Algorithms in the Presence of Registration Errors," *Photogrammetric Engineering & Remote Sensing*, Volume 74, Issue 4, pp.375-383, April 2007

[7] S. Minu, *et al*., "A Comparative Study of Image Change Detection Algorithms in MATLAB," *Aquatic Procedia* , Vol 4, pp 1366–1373, 2015.