Journal of King Saud University – Computer and Information Sciences (2017) **29**, 399–409

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

Multi-scale structural similarity index for motion detection

M. Abdel-Salam Nasr [a], **Mohammed F. AlRahmawy** [b,*], **A.S. Tolba** [b]

[a] *Computer Science Department, Faculty of Computers and Information Sciences, IDA, Ministry of Trade and Industry, Egypt*
[b] *Computer Science Department, Faculty of Computers and Information Sciences, Mansoura University, Egypt*

Received 8 October 2015; revised 9 February 2016; accepted 25 February 2016
Available online 2 April 2016

KEYWORDS

Motion detection;
Multi-scale structural similarity index;
Dynamic template matching

Abstract The most recent approach for measuring the image quality is the structural similarity index (SSI). This paper presents a novel algorithm based on the multi-scale structural similarity index for motion detection (MS-SSIM) in videos. The MS-SSIM approach is based on modeling of image luminance, contrast and structure at multiple scales. The MS-SSIM has resulted in much better performance than the single scale SSI approach but at the cost of relatively lower processing speed. The major advantages of the presented algorithm are both: the higher detection accuracy and the quasi real-time processing speed.

1. Introduction

Fighting terrorism is becoming imperative and is driving researchers toward designing robust video surveillance systems. Event detection and analysis is one of the major applications of video surveillance. A first step in video event analysis is motion detection. A motion detection algorithm should avoid releasing false alarms by considering image luminance, contrast and structure at multiple scales. Accurate motion detection algorithms are the key components of video surveillance systems. It can be applied as well in some interesting applications such as the automatic light control and automatic door opening in smart homes and work areas resulting in efficient energy use.

Intrusion detection in secure areas is a common application of motion detection. A very important application of motion detection is to limit video recording storage requirements to times of important events by starting the recording only at the start of an important event. This on demand recording helps avoiding continuous video recording of static scenes and speeds up video event analysis. Motion detection helps focusing on processing and storage of relevant events.

In this study, a new algorithm is presented which showed promising results compared with MSE (Winkler, 2005; Susstrunk and Winkler, 2004; Wang and Bovik, 2002) and Dynamic Template Matching DTM (Martinez-Martin and del Pobil, 2012a, b; Widyawan et al., 2012) based algorithms. The algorithm captures the video of a scene and detects motion by comparing successive frames in the video sequence based on

[*] Corresponding author.
E-mail addresses: mohtrue.net@gmail.com (M. Abdel-Salam Nasr), mrahmawy@mans.edu.eg (M.F. AlRahmawy), ast@astolba.com (A.S. Tolba).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2016.02.004
1319-1578 © 2016 The Authors. Production and hosting by Elsevier B.V. on behalf of King Saud University.

the modeling of image luminance, contrast and structure at multiple scales.

The paper is organized as follows, in Section 2 some of the related work is presented; then, an overview of the basic and the proposed structural similarity index mechanisms for motion detection is presented. In Section 4, the performance of the proposed mechanism is evaluated using a set of experiments on both offline and online videos. Then, in Section 5, the efficiency of the proposed mechanism is proved by comparing it with the well-known GMM method of motion detection both analytically, by analyzing their memory requirements complexity and execution complexity, and experimentally by comparing their efficiency through a set of performance metrics.

2. Related work

Motion detection starts from a given frame as a reference and subsequent frame is compared with it; then the subsequent frame becomes a reference frame and the process is repeated with next frames. Mishra et al. (2011) discussed three commonly used methods to detect a motion: background subtraction, optical flow and temporal differences. Background subtraction was used by Spagnolo et al. (2006), Tang and Miao (2008), Li and Cao (2010) and it depends on a comparison of an image with a static reference image. The optical flow method specifies how much each pixel of the image moves between adjacent frames, this method may require additional hardware to support the performance and monitoring of systems (Allili et al., 2002; Jung et al., 2007). The temporal difference method relies on comparing consecutive frames by analyzing all frame pixels (Yu and Chen, 2009), e.g. by applying the concept of Sum of Absolute Difference (SAD), where SAD is used to determine whether there is a movement within an image pair (Kenchannavar et al., 2010).

There is a lot of research work based on the above methods and combinations of them. Kenchannavar et al. (2010) described an algorithm combining background subtraction and frame differences for motion detection. Zheng et al. (2009), Murali and Girisha (2009), Fang et al. (2009), used frame differences coupled with an adaptive threshold setting and statistical correlation to analyze the temporal differences in some of the image frames.

Other methods for motion detection exist in the literature. For example, Yong et al. (2011) studied four methods for motion detection: frame differences, background subtraction, pixelate filter, and blob counter. Li and Cao (2010) used Support Vector Machines (SVM) for motion detection and segmentation of moving objects. Yokoyama et al. (2009) applied the concept of vectors to movement detection by comparing multiple frames and marking the points of difference among the frames. An advantage of this method is that it yields information about the direction of object movement. Kameda and Minoh (1996) used the double difference technique for motion detection. Double differences are conducted by comparing two successive frames at times t and $t-1$ and then performing a second comparison between the frames at times $t-1$ and $t-2$. In contrast, Collins and et al. (2000) reported a video surveillance and monitoring system which depends on comparisons between the image t with the image of $t-1$, and the image t with the image of $t-2$.

From the above, we can see that there are many research works done on the motion detection which uses different methods and have achieved different results.

3. Structural similarity index mechanisms for motion detection

The SSI measurement system (Wang and Bovik, 2002) is based on modeling of image luminance, contrast and structure. The MS-SSIM is an extension of the SSI system that achieves better accuracy than the single scale SSI approach but at the cost of relatively lower processing speed. However, the computations required for the MS-SSIM does not require such large computational time required by other efficient Statistical Learning Algorithms (Avcıbas et al., 2002) since it requires complex calculations. Our system applies the MS-SSIM for motion detection in videos; either on online videos directly captured from camera, or on recorded video stored in a file. Any video is sequenced into frames and successive frames are compared with each other and if a difference is detected, an alarm is released.

The proposed approach relies on a new algorithm which helps local adaptation of the multi-scale structural similarity measure for motion detection in videos. In the next sections we present the basic SSI measurement system as it is the base of the used algorithm, the MS-SSIM; next we present MS-SSIM itself. Then, we present our proposed algorithm for motion detection using the MS-SSIM.

3.1. SSI measurement systems

The Structural Similarity Index (SSI) measurement, shown in Fig. 1, is based on modeling image luminance, contrast and structure.

Mathematically, the SSI is defined by Wang and Bovik (2006) as:

$$SSI(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{1}$$

where μ_x, μ_y, σ_x and σ_y are the means and standard deviations of both the original and reference images respectively, C_1 and C_2 are constants. The three models considered in building the similarity index between the two images x and y are given by Lavielle (1999), Wang et al. (2003), and Wang et al. (2004):

$$\text{Luminance}: \ L(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \tag{2}$$

$$\text{Contrast}: \ C(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \tag{3}$$

$$\text{Structure}: \ S(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \tag{4}$$

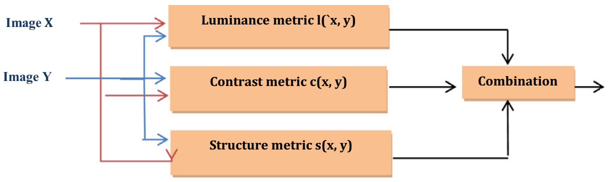


Figure 1 Structure similarity measurement system (Tolba and Raafat, 2015).

where μ_x, $\sigma^2{}_x$ and σ_{xy} are the mean of x, the variance of x, and the covariance of x and y respectively, while C_1, C_2 and C_3 are constants given by $C_1 = (K_1L)^2$, $C_2 = (K_2L)^2$ and $C_3 = C_2/2$. L is the dynamic range for the sample date, where, $L = 255$ for 8 bit gray-level image and $K_1 \ll 1$ and $K_2 \gg 1$ are two scalar constants (Winkler, 2005; Susstrunk and Winkler, 2004). Given the above measures the structural similarity index of two images x and y can be computed as (Winkler, 2005; Susstrunk and Winkler, 2004; Lavielle, 1999; Wang et al., 2003, 2004):

$$ssi(x,y) = [l(x,y)]^{\alpha}[c(x,y)]^{\beta}[s(x,y)]^{\gamma} \tag{5}$$

where α, β, γ define the weight given to each model. Fig. 2 shows the architecture of a motion detection system that is based on the SSI.

3.2. MS-SSIM measurement systems

The Multi Scale Structural Similarity Index for Motion Detection (MS-SSIM) quality metric is an extension of the SSI which computes these measures at various scales and combines them using an equation of the form (Lavielle, 1999; Wang et al., 2003, 2004):

$$MS-SSIM(x,y) = [l_m(x,y)]^{\alpha_M}.\prod_{j=1}^{M}[C_j(x,y)]^{\beta_j}.[S_j(x,y)]^{\gamma_j} \tag{6}$$

where, M corresponds to the lowest resolution (i.e. the times of down samplings performed to reduce the image resolution), while $j = 1$ corresponds to the original resolution of the image. The resolution of the analyzed image has a significant impact on motion detection results. The interaction between motion size and image resolution also is an important factor. Therefore, using the MS-SSIM metric renders itself a good adaptive measure for motion detection. Fig. 3 presents the architecture of a novel system for Motion detection based on the multi-scale structural similarity index.

3.3. Proposed MS-SSIM based motion detection algorithm

The main steps of the proposed algorithm to detect motion in a video sequence using the MS-SSI, are shown in Fig. 4 and are summarized as follows:

1- The algorithm starts by loading the video file in case of a recorded video, or initializing video capturing through the camera in case of real time motion detection.

2- Then, the following steps are applied repeatedly on the successive frames of the captured video starting from time $t = 0$, i.e. first captured/loaded frame; until, in case of a recorded video, all video frames have been processed; or, in case of real time video, the camera stops capturing the scene:

a. Capture/load one frame f_t at the time instant t.
b. Capture/load the next frame f_{t+1} at time instant $t + 1$.
c. Convert the two frames to gray level format.
d. Compute the MS-SSIM (f_t, f_{t+1}) index of the two successive frames.
e. If the similarity between the two successive frames lies below a predetermined threshold θ a motion activity is detected and an alarm is released.

In the above algorithm, the estimation of the appropriate threshold level for the MS-SSIM test is very critical to the success of the presented detection system.

4. Performance evaluation

In the following subsections, we present the results of applying the proposed algorithm in different cases on different video samples.

4.1. Test results of recorded video with predetermined number of motions

In the following test, the performance of the developed algorithm is evaluated using recorded videos. The changes between frames have been identified to detect the occurrence of a motion. To measure the performance for the proposed motion detection approach, we have adapted some performance measures discussed by Altman and Bland (1994), Lu and et al. (2004), Eisner and et al. (2005), and Sokolova et al. (2006), Gonzàlez et al. (2007). These measures are defined in our work as:

True Still Stand (TSS): Number of two successive frames where no motion is detected based on the agreement of both the system result and the ground truth.

False Still Stand (FSS): Number of two successive frames where no motion is detected while there is disagreement between the system result and the ground truth.

True Motion Detection (TMD): Number of two successive frames where motion is detected based on the agreement of both the system result and the ground truth.

False Motion Detection (FMD): Number of two successive Frames where motion is detected while there is disagreement between the system result and the ground truth.

F-Score: A measurement of the accuracy that considers both precision and sensitivity of the test to compute the score.

Matthews' Correlation Coefficient (MCC): A balanced measurement of the quality of binary classifications i.e. it is a correlation coefficient between the observed and the predicted binary classifications, where positive 1 value indicates perfect prediction, 0 indicates random average prediction and −1 value indicates inverse prediction.

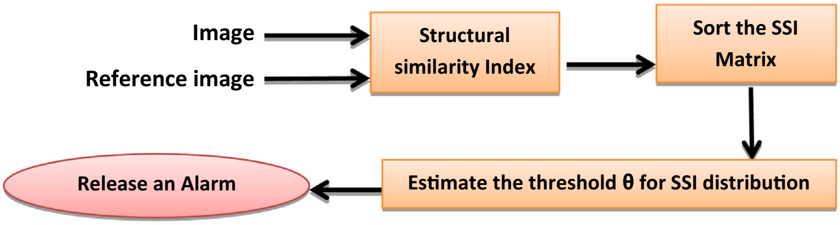


Image
Reference image
Structural similarity Index
Sort the SSI Matrix
Estimate the threshold θ for SSI distribution
Release an Alarm

Figure 2 Architecture of the SSI based motion detection system [adapted from (Tolba and Raafat, 2015)].

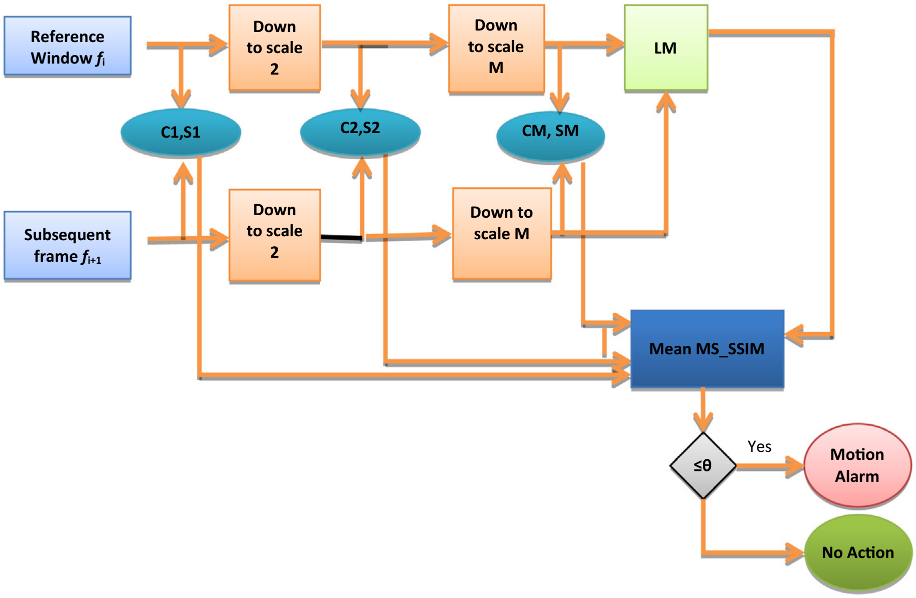


Figure 3 Architecture of the MS-SSIM based motion detection system.

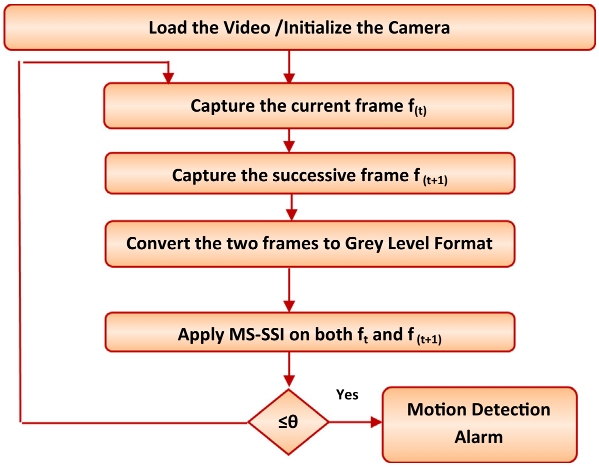

Load the Video /Initialize the Camera
Capture the current frame $f_{(t)}$
Capture the successive frame $f_{(t+1)}$
Convert the two frames to Grey Level Format
Apply MS-SSI on both f_t and $f_{(t+1)}$
≤θ
Yes
Motion Detection Alarm

Figure 4 MS-SSIM based motion detection system.

From these measures, we used the following performance indicators of the motion detection system:

$$precision = \frac{TMD}{TMD + FMD} \tag{7}$$

$$Sensitivity = \frac{TMD}{TMD + FSS} \tag{8}$$

$$Specificity = \frac{TSS}{TSS + FMD} \tag{9}$$

$$Accuracy = \frac{TMD + TSS}{TMD + TSS + FMD + FSS} \tag{10}$$

$$Positive\ Prediction = \frac{TMD}{TMD + FMD} \tag{11}$$

$$Negative\ Prediction = \frac{TSS}{TSS + FSS} \tag{12}$$

$$False\ Negative\ Rate = \frac{FSS}{FSS + TMD} \tag{13}$$

$$False\ Positive\ Rate = \frac{FMD}{FMD + TSS} \tag{14}$$

$$F - score = \frac{(2 \times Precision \times Sensitivity)}{(Precision + Sensitivity)} \tag{15}$$

$$MCC = \frac{(TMD * FMD) - (TSS * FSS)}{\sqrt{((TMD + FSS) * (TMD + TSS) * (FSS + TMD) * (FSS + FMD))}} \tag{16}$$

Furthermore, we checked the quality of the proposed algorithm at different threshold values to measure the effect of the chosen threshold on the correctness of the algorithm.

In the experiments, we used three videos with predetermined motion actions (ground truth), where each of these videos shows a clock with pointer(s) moving at a constant rate. The choice of such videos simplifies the visual detection of the motion occurrence and limits it to pre-determined values that can be easily compared with the motion detected by the proposed algorithm to measure the accuracy and efficiency of the proposed algorithm at different threshold values. The snapshot of the first video shown in Fig. 5, has a clock with a single *second pointer*, i.e. moving at a rate of 60 movements/minute. The video specifications are,

Length of video: 60 s. Number of frames: 615 frames.
Frame size: 640 * 360 size: 44.9 MB (Uncompressed).
Frame rate 10 f/s.

From the above details, we can easily conclude that the video has 60 movements. By applying the proposed algorithm on this video at different threshold values we obtained the results shown in Table 1. From the results shown in the table, we see that the detection rate of the algorithm increases with increasing value of the threshold as expected. The results in Table 2 show the calculated performance measurements defined earlier, these measurements show that a threshold value of less than 0.975 can be considered non-practical due to the very low values of sensitivity and the relatively low values of precision. Also, we note that the number of false detected movements (FMD) increased by increasing the threshold value above 0.985. This is expected, in this video, as the movement of the *seconds' pointer* from a position to another at some moments, in the video, causes the *seconds' pointer* to be drawn twice at two different positions at the same time for a short period (less than a second), which causes the proposed algorithm to detect the same movement more than once at very high threshold values. The increase in the number of the false detected movements results in a decrease in the specificity, the accuracy and the precision of the proposed algorithm for threshold values above 0.985.

The algorithm in case of the first video was applied on a video with a high quality. To test the effect of the quality of the video on choosing the value of the threshold, the algorithm has been applied on a second video that has a clock with three pointers and it has a low quality, see Fig. 6. The video has the following specifications:

Duration of video: 00:01:00
Number of frames: 602 frames.
Frame size: 360 * 640.
Frame rate 10 f/s.

From the characteristics of the second video, we can see that the video should have 61 movements; one movement for the *minutes' pointer* and 60 movements for the *seconds' pointer*. Tables 3 and 4 shows the results of applying the proposed algorithm on this video. For this video, using threshold values less than 0.980 is not practical. But the detection of false movements, unlike the first video, does not occur until a threshold value of about 0.995.

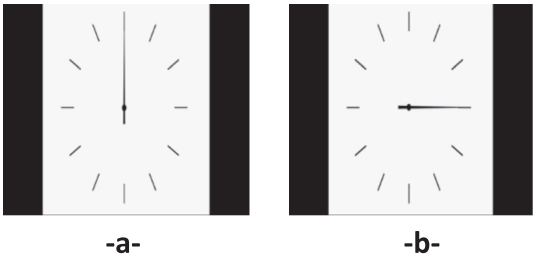


Figure 5 Snapshot of a video with known number of motions.

Table 1 Results of applying the proposed algorithm on the first video.

	θ							
	0.96	0.965	0.97	0.975	0.98	0.985	0.99	0.995
TMD	1	1	29	45	60	60	60	60
TSS	552	552	553	553	553	553	552	537
FMD	2	2	2	2	2	2	3	18
FSS	60	60	31	15	0	0	0	0

For this video, the accuracy was about 0.05 only at θ = 0.985 and increased to 0.67 at θ = .990 and reached .934 at θ = 0.995. However, unlike the case of the previous video, the algorithm has scored only 5 false motion detections at θ = 1 (not shown in the table) and not detected any false movements up to θ = 0.995. This accounts for the constant values for both the *false positive rate* and the *positive prediction* in Table 4. This is because the movement of both the *minutes' pointer* and the *seconds' pointer* in this video was sharp, i.e. there is no clearly observed repeated drawing of the same pointer in a single movement as it happens in the previous video.

From the above experiments, it can be seen clearly that the right choice of the value of θ can be affected by the quality of the video, where for good quality videos a value between 0.985 and 0.990 gives the best performance measurements, whereas for less quality videos this value should increase up to 0.995 to achieve the best results. Also, the sensitivity of the algorithm to detect false movements (or detect the same movement more than once) increases when the threshold value is 0.995 or above.

4.2. Test results of recorded video with continuous motion

The aim of this experiment is to test the efficiency of the algorithm in detecting the motions in a recorded video at different threshold and frame rate values.

Due to the continuous motion, in the video, it is difficult to calculate an exact number of frames that have motions. So, we adopted a manual approach to measure the performance by observing the scenes of the video to detect the time interval (s) that included a motion, i.e. the region of interest (ROI). Then, we manually can deduce an estimated number of frames that has a motion; *nef*, by multiplying the summation of these time intervals and multiply the results by the frame rate to get

Table 2 Performance measurements of applying MS-SSIM on the first video.

Θ	Sensitivity	Specificity	Accuracy	Precision	Positive prediction	Negative prediction	False negative rate	False positive rate
0.970	0.08	0.90	0.907	1.00	0.935	0.950	0.517	0.004
0.975	0.54	0.95	0.954	1.00	0.957	0.974	0.250	0.004
0.980	0.77	1.00	0.977	1.00	0.968	1.000	0.000	0.004
0.985	0.93	1.00	0.993	1.00	0.968	1.000	0.000	0.004
0.990	0.98	1.00	0.998	1.00	0.952	1.000	0.000	0.005
0.995	1.00	1.00	1.000	1.00	0.770	1.000	0.000	0.032
Average [0.980:0995]	0.92	1.00	0.992	1.00	0.915	1.000	0.000	0.011

an estimated number of the number of frames that has a motion. Then, at different values of the threshold and frame rate, we compare the actual number of frames detected by the proposed system; *ndf*, to the numbers of estimated frames (*nef*). Also, another number of frames; *nof*, is counted by observing the same video and this value is compared as well to the values detected by the MS-SSI at different threshold and frame rate values.

Fig. 7 shows snapshots of a video captured in a restaurant using a static camera to monitor persons in the restaurant. The experiment is performed on a test video taken from the Image Sequence Evaluation (ISE) Lab (Gonzàlez et al., 2007), see a sequence of it in Fig. 7. The original video specifications are as follows:

Duration of video: 00:02:13
Number of frames: 3321 frames.
Frame size: 1392 px, Height 1040 px,
File format: AVI.
Frame rate: 25 fps.

To get the frames' ROI in this video, it is found that there was no motion in the first 24 s; then, the persons started to appear and move in the scene from the moment at 00:24:00 up to the moment 01:36:00. In addition to that, there was a short moment of motion in other part of the video for less than one second. So, a total of around 73.5 s was assumed as the length of the ROI of the frames. This value was used to get the estimated number of frames in the ROI at three different fame rate values, 25, 18, 10 f/s, as shown in Table 5. The same table also shows the counted number of frames, *nof,*

Table 6 shows the number of the detected frames (*ndf*) detected by the proposed MS-SSI approach at several values of the threshold obtained manually by observing the video. The results of the two Tables 5 and 6 are used to draw the charts shown in Fig. 8 in order to analyze the efficiency of the MS-SSI with the change of the threshold value used in the algorithm and the frame rate of the analyzed video.

Fig. 8(a) shows clearly that the detection rate increases with the increase in the value of the threshold as expected from the previous experiments. Fig. 8(b) shows an interesting anomaly, where the detected number of frames, at any value of the threshold, for the 18 f/s is more than the detection rate of the 10 f/s but it is less than the 25 f/s. This is due to the fact that increasing the frame rate from 10 f/s to 18 f/s results in an increase in the number of frames of the video; and hence the number of frames in the frames of the period of the motion, which makes the MS-SSI able to detect higher number of frames. However, continuous increase in the value of the frame rate makes the differences between each two consecutive frames less noticeable by the algorithm and requires the use of a higher threshold value of the algorithm to be detected; hence, resulting in a decrease in the number of the detected frames of motion. Fig. 8(c and f) gives a better comparison by avoiding variation of the numbers of frames in the three versions of the video, by normalizing the number of detected frames to the number of estimated and observed frames, i.e. *ndf*/*nef* and *ndf*/*nof* respectively in the period of interest.

Fig. 8(c) shows clearly that the detection efficiency increases with the increase in the threshold, where the detection reaches high values for 10 f/s than the cases of 18 f/s and 25 f/s, i.e. increasing the frame rate requires the use of higher threshold values to get higher detection efficiency. The same Figure shows that a threshold value of 0.99 results in the detection of a ratio around 100% for the 10 f/s, but using this value in case of the 18 f/s results in 69% and only 50% in case of 25 f/s. Fig. 8(d), shows this clearly, as the chart lines are monotonically decreasing with a sharp slope, i.e. the increase in the frame rate results in lower detection efficiency of the MS-SSI. The low detection ratios of Fig. 8(c and d) may indicate inefficiency of the proposed MS-SSI at high frame rates, but looking at Fig. 8(e) gives a better indication as it shows the ratio of the detected motion by the MS-SSI to the number of motion frames *observed* by humans, which is more sensible.

Fig. 8(e) shows that the MS-SSI resulting motion detection for both 10 f/s and 18 f/s are very close to each other for all values of the threshold, while for the 25 f/s it was a bit less for low threshold values until the threshold value of 0.985, where its tarts to be very close to the 10 f/s and the 18 f/s and at a threshold value of 0.99 it has made a 0.98% detection success. This is again due to the fact that using high frame rate

Table 3 Measurements of the detected frames in the second video.

	Θ							
	0.96	0.965	0.97	0.975	0.98	0.985	0.99	0.995
TMD	0	0	5	33	47	57	60	61
TSS	544	544	544	544	544	544	544	544
FMD	0	0	0	0	0	0	0	0
FSS	61	61	56	28	14	4	1	0

Table 4 Performance metrics obtained by applying MS-SSIM on the second video at different threshold values.

Θ	Sensitivity	Specificity	Accuracy	Precision	Positive prediction	Negative prediction	False negative rate	False positive rate
0.960	0.016	0.901	0.899	0.333	–	0.899	1.00	0.000
0.965	0.016	0.901	0.900	0.333	–	0.899	1.00	0.000
0.970	0.483	0.946	0.946	0.935	1.00	0.907	0.918	0.000
0.975	0.750	0.996	0.972	0.957	1.00	0.951	0.460	0.000
0.980	1.000	0.996	0.997	0.970	1.00	0.975	0.230	0.000
0.985	1.000	0.996	0.996	0.970	1.00	0.993	0.066	0.000
0.990	1.000	0.995	0.995	0.952	1.00	0.998	0.016	0.000
0.995	1.000	0.968	0.970	0.770	1.00	1.000	0.000	0.000
Average [0.975:0.985]	0.917	99.61	0.988	0.966	1.000	0.973	0.450	0.000

Figure 6 Snapshot of the second video with known number of motions.

results in consecutive frames with very similar structure that requires high threshold values to be detected by the MS-SSI. The same result is deduced from Fig. 8(f); at low threshold values, the chart lines decrease with the increase in the frame rate, while at high threshold values the lines look constant; i.e. the ratio *ndf*/*nof* is constant versus the frame rate change. From the above, it is seen that the value of the threshold should be taken between 0.985 and 0.995, with a recommendation of using values closer to 0.995 for the videos with a high frame rate.

4.3. Results of motion detection in real time video

Real time videos captured using a webcam has been used to test the algorithm. The frames of the video sequence were captured at a resolution of 192 × 256 and converted to gray level. The sequence included 1236 frames. The MS-SSIM algorithm has succeeded in motion detection with a true positive rate of 99.1% at a threshold value of 93%. The DTM algorithm resulted in only 89.9% correct detection rate. This clearly shows the efficiency of the proposed method relative to the classical methods of motion detection.

5. Efficiency evaluation

As stated earlier, existing techniques for motion detection include methods based on optical flow, frame difference, template matching and background subtraction methods. Among those, Background subtraction methods have been proven to be the most efficient as indicated by Sun and et al. (2004), Xie, 2013, and Benezeth et al. (2010). Hence, in this section, we evaluate the efficiency of our proposed MS-SSI based motion detection algorithm by comparing it with one of the most efficient background subtraction methods (Benezeth et al., 2010); the Gaussian Mixture Model (GMM). The main idea of the GMM method is to use multimodal probability density functions, where every pixel within a frame is modeled with a mixture of K Gaussians; hence, the probability of occurrence of a color $p(I_{s,t})$ at a given pixel s in frame t is given by:

$$p(I_{s,t}) = \sum_{i=1}^{K} w_{i,s,t} N\left(\mu_{i,s,t}, \sum_{i,s,t}\right) \tag{17}$$

where $N(\mu_{i,s,t}, \sum_{i,s,t})$ is the ith Gaussian model, $w_{i,s,t}$ is its weight, $\mu_{i,s,t}$ is the ith Gaussian mean and $\sum_{i,s,t}$ is the covariance matrix. The weight $w_{i,s,t}$ of each distribution is updated for each frame according to the equations in Benezeth et al. (2010).

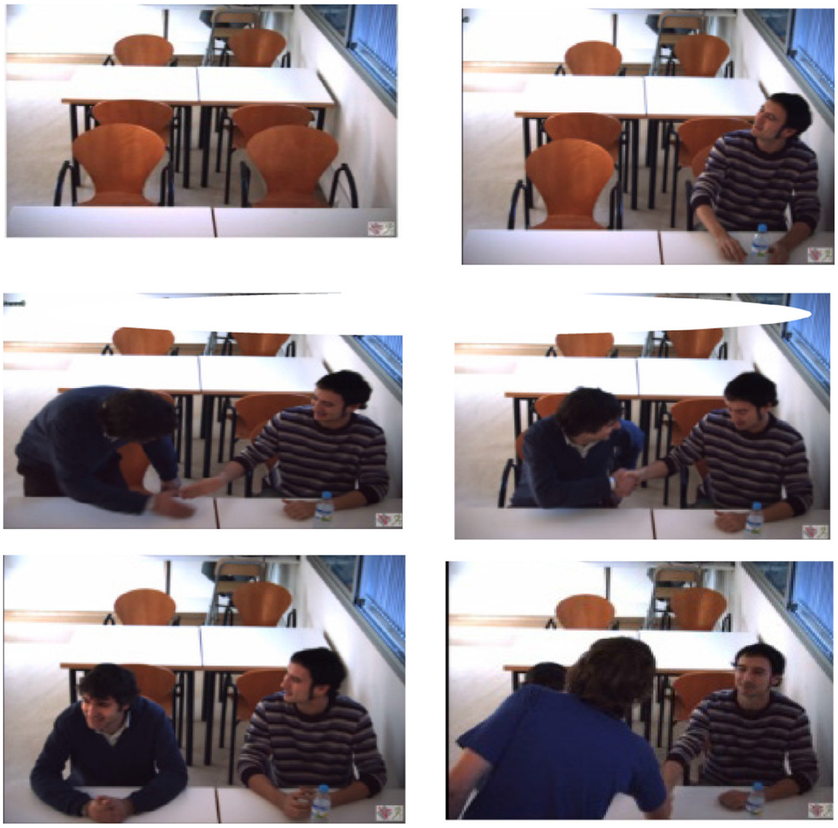


Figure 7 Snapshot of the video with continuous motion.

Table 5 Average system performance indicators for MS-SSIM when applied on the video shown in Fig. 7.

	Observed movements at frame rate		
	10 f/s	18 f/s	25 f/s
Total number of frames in the video	1331 frames	2394 frames	3321 frame
Estimated frames in the ROI (*nef*)	738 frames	1324 frame	1833 frame
Observed frames of motion (*nof*)	738 frames	888 frames	918 frames

Table 6 Measurements of the detected frames in the video shown in Fig. 7.

Threshold value	Total detected frames (*ndf*) of movements at		
	10 f/s	18 f/s	25 f/s
0.930	348	382	316
0.940	381	433	360
0.950	434	502	404
0.960	510	572	474
0.965	510	572	474
0.970	571	655	581
0.975	571	655	581
0.980	626	763	715
0.985	626	763	715
0.990	760	910	908
0.995	760	910	908

In the following, we compare the proposed method with the GMM method in three main aspects: the memory requirements, computation time complexity and the accuracy.

5.1. Memory requirements analysis

For the basic GMM model, the memory storage space required is 5*K* floats per pixel (Benezeth et al., 2010), where *K* is the number of distributions built per pixel and *K* is practically taken between 3 and 5, this relatively big storage area is due to the need for representing every pixel with *K* probability density function. On the contrary, our proposed method, which is based on the MS-SSIM, computes at various scales from 1 down to *M*, the contrast and the structure terms; in addition to the luminance at the lowest scale only. Hence, the main storage S_{frame} required for storing and processing the pixels in a single video frame processed using this method can be calculated by summing up the sizes required for processing all the pixels in this frame at all the scales from 1 to *M* as follows:

$$S_{frame} = N_{pixels[1...M]}.S_{cs-pixel} + N_{(M)}.S_l \tag{18}$$

where, $N_{pixels[1...M]}$ is the total number of pixels processed at all scales from 1 to *M*, $S_{cs\text{-}pixel}$ is the storage size required for processing one pixel to get both its structure and contrast values at a certain scale, N_M is the number of pixels at scale *M* and S_{l} is the storage required for luminance.

$N_{pixels[1...M]}$ can be calculated as follows:

$$N_{pixels[1...M]} = S_{ScaleF(1)} + S_{ScaleF(2)} + \cdots S_{ScaleF(M)} \tag{19}$$

where $S_{\mathrm{Scale}F(i)}$ is the number of pixels in frame *i*; hence, $N_{pixels[1...M]} = (1 + (1/(2.2)) + \cdots + (1/(M.M))).\ S_{ScaleF(1)}$ It is easily to conclude that,

$$N_{pixels[1...M]} < 2.S_{ScaleF(1)}, \tag{20}$$

where $S_{ScaleF(1)}$ is frame size at scale 1. Hence, from the above and as $N_M = S_{ScaleF(M)} < S_{ScaleF(1)}$

$$S_{frame} < 2.S_{ScaleF(1)}.S_{cs-pixel} + S_{ScaleF(1)}.S_l \tag{21}$$

Hence, we can conclude that the storage required to calculate the MS-SSI of a single pixel $S_{MS\text{-}SSI\text{-}pixel}$ can be given by $S_{MS\text{-}SSI\text{-}pixel} < (2.S_{ScaleF(1)}.\ S_{cs\text{-}pixel} + S_{ScaleF(1)}.\ S_l)/S_{ScaleF(1)}$; hence,

$$S_{MS-SSI-pixel} < 2.S_{cs-pixel} + S_l \tag{22}$$

In MS-SSI, to calculate both $S_{cs\text{-}pixel}$ and S_l for each pixel, a 11 × 11 window is typically used at each pixel, to get both the contrast and the structure values as indicated in equations (Susstrunk and Winkler, 2004; Wang and Bovik, 2002; Martinez-Martin and del Pobil, 2012a, b) then the same window moves to the next pixel. Hence, only 2 floats are required to save both the structure and the contrast at each pixel, i.e. $S_{cs\text{-}pixel} = 2$, and only one float is required for the luminance value, i.e. $S_l = 1$. Hence, by substituting in the above equation, $S_{MS\text{-}SSI\text{-}pixel} < 5$, i.e. the proposed method which is based on the MS-SSIM requires less than 5 floating point operations for processing each pixel in a video frame, which is K times less storage than the GMM-based method.

5.2. Time complexity analysis

For the computation time, the GMM is much more complicated and consumes more time as it involves several stages including:

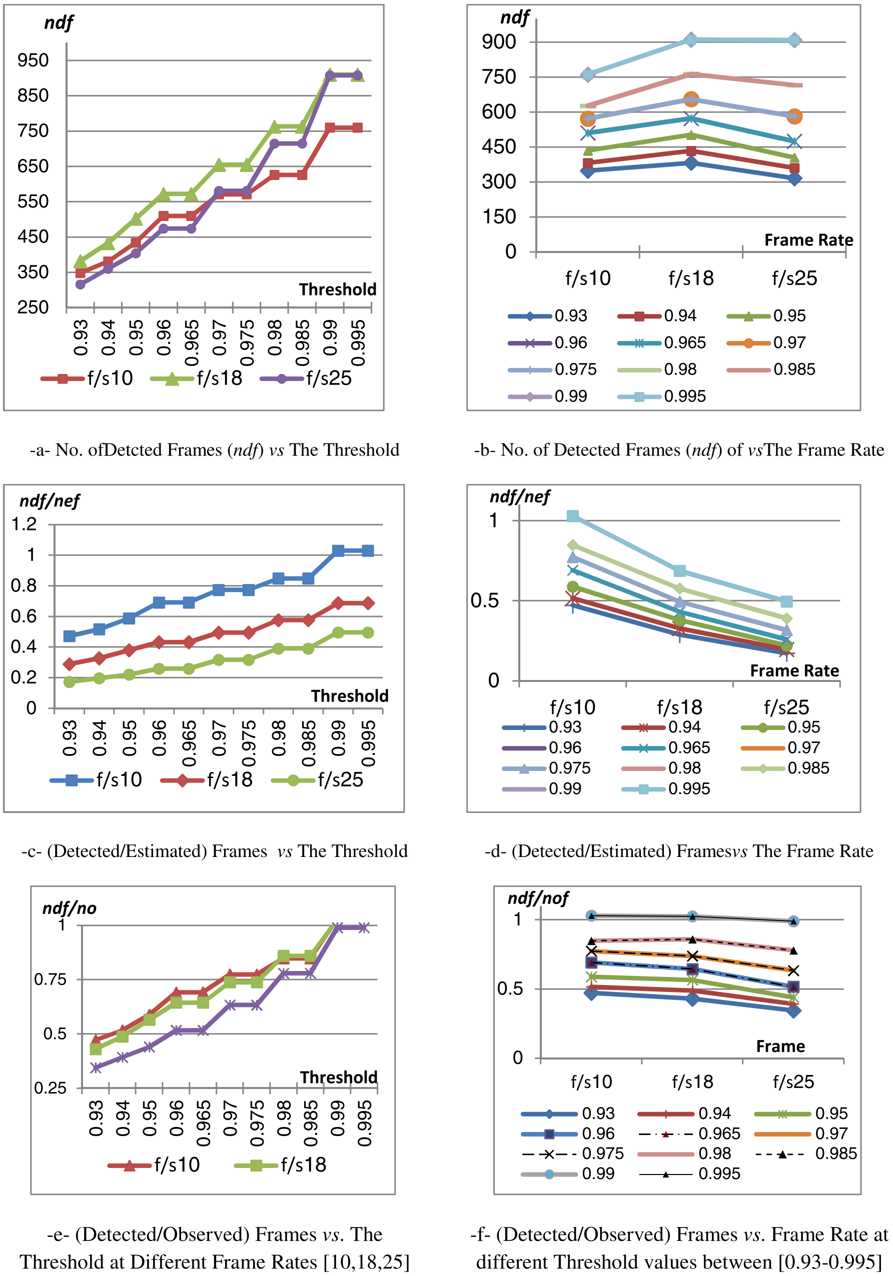
- Calculate *K* Gaussian distributions for each pixel in each frame.
- Update the weights of the distributions from a frame to another.
- Order the distributions and extract the best background model for each frame.
- Segment the foreground and detect the motion by subtracting it from the foreground of the previous frame.

All these operations make GMM much more time consuming when compared to the proposed MS-SSIM. MS-SSIM requires processing a number of pixels $N_{pixels[1...M]} < 2.\ S_{ScaleF(1)}$, where it requires for each pixel only one distribution to use it with the distribution of the pixel at the same location in the following frame to calculate both the structure and the contrast; then, the luminance is calculated only once on the last scaled down image. Hence; the time complexity to process the pixels in one video frame using our MS-SSIM based method is given by:

$$T_{single-frame} = N_{pixels[1...M]}.T_{sc} + N_M.T_l \tag{23}$$

However, Both T_{sc} and T_l are constants as they use a fixed 11 × 11 window. Hence,

$$\begin{aligned} T_{single-frame} &= c_1.N_{pixels[1...M]} + c_2.N_M \\ &= O(N_{pixels[1...M]}) \text{as } N_M \text{ is much less than} N_{pixels[1...M]} \end{aligned}$$



-a- No. ofDetcted Frames (*ndf*) *vs* The Threshold

-b- No. of Detected Frames (*ndf*) of *vs*The Frame Rate

-c- (Detected/Estimated) Frames *vs* The Threshold

-d- (Detected/Estimated) Frames*vs* The Frame Rate

-e- (Detected/Observed) Frames *vs*. The Threshold at Different Frame Rates [10,18,25]

-f- (Detected/Observed) Frames *vs*. Frame Rate at different Threshold values between [0.93-0.995]

Figure 8 Effect of threshold and the frame rate on the MS-SSSI.

As $N_{pixels[1\ldots M]} < 2.S_{ScaleF(1)}$, hence, the MS-SSIM requires a number of iterations less than twice the video frame size to detect the motion. In contrast to GMM, to run the first stage, i.e. to get the K distributions of each pixel requires a number of times equals K times the frame size. This means the whole processing of the proposed MS-SSIM is faster than the first stage of the GMM-based method, which clearly emphasizes the efficiency of the proposed method.

5.3. Performance of detection

To compare their performance, both the GMM method and the proposed method have been tested on the videos shown earlier in Figs. 5–7. Table 7 explores the average performance

Table 7 Comparison between GMM and the proposed method.

Performance Indicators	MSSI-MD	GMM
Sensitivity	92%	93%
Specificity	100%	99.8%
Accuracy	99.2%	99.1%
Precision	100%	98.3%
F-score	0.9583	0.9558
MCC	0.17	−0.09
Positive-prediction	91.5%	98.3%
Negative-prediction	100%	99.6%
False-negative-rate	0	3.3%
False-positive-rate	1.1%	0.18%

measurements of both methods on both videos. It is clear that the proposed method provides very high specificity, accuracy and precision in the detection with very high sensitivity, F-Score, a positive MCC and very low false rates in comparison with the GMM.

6. Conclusions

This work presents an efficient novel approach for detection of motion occurrence within videos. It measures the similarity of successive frames of a video using the Multi Scale Structural Similarity Index. The proposed approach has achieved very high motion detection accuracy in most of the conducted experiments; its efficiency depends on the used threshold, where it gives the best results for values between 0.985 and 0.995 in most experiments. The experimental results show that the proposed MS-SSIM-based method has resulted in better performance than the single scale SSI approach but at the cost of relatively lower processing speed, this reduction in speed is expected as it applies the SSI on several scales. In comparison, with existing high efficient methods, the proposed method has shown to provide high efficiency with high speed and relatively low storage requirements which make it a great candidate for use in embedded devices with limited resources. The major advantages of the presented approach are: the higher detection accuracy and the quasi real-time processing speed. The resilience of the proposed approach to luminance, contrast and structural changes makes it very suitable for motion detection applications. The proposed approach at the moment detects the motion occurrence not the moving objects; in the future, we plan to integrate it with other existing approaches for detecting the moving objects. Also, further research will focus on the area of integrating the developed MS-SSIM into embedded systems with other non-camera based motion detection approaches to build multi-sensing modalities for motion detection.

Acknowledgement

We thank our colleague Dr. Ehab Essa for his help and support in reviewing this work and for his helpful comments and advices to finish this work.

References

Allili, M., Auclair-Fortier, M.-F., Poulin, P., Ziou, D., 2002. A computational algebraic topology approach for optical flow. In: ICPR '02 Proceedings of the 16th International Conference on Pattern Recognition (ICPR'02) Volume 1 – Volume 1, Washington DC, USA.

Altman, D.G., Bland, J.M., 1994. Statistics notes: diagnostic tests 1: sensitivity and specificity. BMJ 308 (1552).

Avcıbas, I., Sankur, B., Sayood, K., 2002. Statistical evaluation of image quality measures. J. Electron. Imaging 11 (2), 206–223.

Benezeth, Y., Jodoin, P.-M., Emile, B., Laurent, H., Rosenberger, C., 2010. Comparative study of background subtraction algorithms. J. Electron. Imaging 19 (3), 1–12.

Collins, Robert T. et al, 2000. A System for Video Surveillance and Monitoring. Carnegie Mellon University, Robotics Institute.

Eisner, R. et al, 2005. Improving protein function prediction using the hierarchical structure of the gene ontology. In: IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology.

Fang, Li, Meng, Zhang, Chen, Claire, Hui, Qian, 2009. Smart motion detection surveillance system. In: 2009 International Conference on Education Technology and Computer, Singapore, pp. 171–175.

Gonzàlez, Jordi, Xavier Roca, F., Villanueva, Juan José, 2007. HERMES: a research project on human sequence evaluation. In: Computational Vision and Medical Image Processing (VipIMAGE'2007), Porto, Portugal.

Jung, Ho Gi, KyuSuhr, Jae, Bae, Kwanghyuk, Kim, Jaihie, 2007. Free parking space detection using optical flow-based euclidean 3D reconstruction. In: Proceedings of the IAPR Conference on Machine Vision Applications (IAPR MVA 2007), Tokyo, Japan, pp. 16–18.

Kameda, Y., Minoh, M., 1996. A human motion estimation method using 3-successive video frames. In: International Conference on Virtual Systems and Multimedia, pp. 135–140.

Kenchannavar, H.H., Patkar, Gaurang S., Kulkarni, U.P., Math, M. M., 2010. Simulink model for frame difference and background subtraction comparison in visual sensor network. In: 2010 The 3rd International Conference on Machine Vision (ICMV 2010), Hongkong China.

Lavielle, M., 1999. Detection of multiple changes in a sequence of dependent variables. Stochastic. Processes Appl. 83 (2), 79–102.

Li, Hongyan, Cao, Hongyan, 2010. Detection and segmentation of moving objects based on support vector machine. In: 2010 Third International Symposium on Information Processing, Shandong China, pp. 193–197.

Lu, Z. et al, 2004. Predicting subcellular localization of proteins using machine-learned classifiers. Bioinformatics 20 (4), 547–556.

Martínez-Martín, E., del Pobil, A.P., 2012a. Robust motion detection in real-life scenarios. Springer, Berlin/Heidelberg, Germany, pp. 16–18.

Martinez-Martin, E.., del Pobil, A.P., 2012b. Robust Motion detection in real-life scenarios, Springer Briefs in computer science, @EasterMartinez.Martin 2012. doi:http://dx.doi.org/10.1007/978-1-4471-4216-4_1.

Mishra, Sumita, Mishra, Prabhat, Chaudhary, Naresh K., Asthana, Pallavi, 2011. A novel comprehensive method for real time video motion detection surveillance. Int. J. Sci. Eng. Res. 2 (4).

Murali, S., Girisha, R., 2009. Segmentation of motion objects from surveillance video sequences using temporal differencing combined with multiple correlation. In: 2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance, Genova, Italy, pp. 472–477.

Sokolova, M., Japkowicz, N., Szpakowicz, S., 2006. Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation. In: Australian Conference on Artificial Intelligence, vol. 4304, pp. 1015–1021, LNCS, Germany.

Spagnolo, P., D'Orazio, T., Leo, M., Distante, A., 2006. Moving object segmentation by background subtraction and temporal analysis. Image Vis. Comput. 24, 411–423.

Sun, Y. et al, 2004. From GMM to HGMM: an approach in moving object detection. Comput. Inf. 23, 215–237.

Susstrunk, S., Winkler, S., 2004. Color image quality on the internet. Proc. SPIE Electronic Imaging: Internet Imaging 5304, 118–131.

Tang, Zhen, Miao, Zhenjiang, 2008. Fast Background Subtraction Using Improved GMM and Graph Cut. In: Congress on Image and Signal Processing, CISP '08, pp. 181–185.

Tolba, A.S., Raafat, Hazem M., 2015. Multiscale image quality measures for defect detection in thin films Springer, London. Int. J. Adv. Manuf. Technol. 79 (1–4), 113–122.

Wang, Z., Bovik, A.C., 2002. Why is image quality assessment so difficult? Proc. IEEE Int. Conf., Acoustics, Speech and Signal Processing 4, 3313–3316.

Wang, Z., Bovik, A.C., 2006. Modern Image Quality Assessment. Morgan and Claypool Publishers.

Wang, Zhou, Simoncelli, Eero P., Bovik, Alan C., 2003. Multi-scale structural similarity for image quality assessment. In: Proc. of the 37th IEEE Conference on Signals, Systems and Computers, Pacific Grove, CA, Nov. 9-12.

Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P., 2004. Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Processing 13 (4), 600–612.

Widyawan, Muhammad, IhsanZul, 2012. Adaptive Motion Detection Algorithm using Frame Differences and Dynamic Template Matching Method. In: The 9th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 2012), Nov. 26–28, 2012 in Daejeon Convention Center (DCC), Daejeon, Korea.

Winkler, S., 2005. Digital Video Quality: Vision Models and Metrics. John Wiley & Sons Ltd, West Sussex, England.

Xie, Y., 2013. Improved gaussian mixture model in video motion detection. J. Multimedia 8 (5), 527–533.

Yokoyama, Takanori, Iwasaki, Toshiki, Watanabe, Toshinori, 2009. Motion vector based moving object detection and tracking in the MPEG compressed domain. In: 2009 Seventh International Workshop on Content-Based Multimedia Indexing, Chania, Crete, pp. 201–206.

Yong, Yee Ching, Sudirman, Rubita, MeyChew, Kim, 2011. Motion detection and analysis with four different detectors. In: 2011 Third International Conference on Computational Intelligence, Modelling and Simulation, Langkawi, pp. 46–50.

Yu, Zhen, Chen, Yanping, 2009. A real-time motion detection algorithm for traffic monitoring systems based on consecutive temporal difference. In: Proceedings of the 7th Asian Control Conference, Hong Kong, China, August 27–29.

Zheng, Xiaoshi, Zhao, Yanling, Li, Na, Wu, Huimin, 2009. An automatic moving object detection algorithm for video surveillance applications. In: 2009 International Conference on Embedded Software and Systems, Hangzhou China, pp. 541–543.