UDC 004.41

Building an Advanced Invariant Real-Time Human Tracking System

Fayez Idris[1], Mazen Abu_Zaher[2], Rashad J. Rasras[3], and Ibrahiem M. M. El Emary[4]

[1] School of Informatics and Computing, German-Jordanian University
fayez.idris@gju.edu.jo

[2] Faculty of Graduate Students, Jordan University of Science and Technology
mazen_mmaz@yahoo.com

[3] Faculty of Engineering Technology, Albalqa' Applied university, Amman, Jordan
rashad_ras@yahoo.com

4 Faculty of Engineering, Al Ahliyya Amman University, Amman, Jordan
doctor_ebrahim@yahoo.com

Abstract. Real-time human tracking is very important in surveillance and robot applications. We note that the performance of any human tracking system depends on its accuracy and its ability to deal with various human sizes in a fast way. In this paper, we combined the presented works in [1, 2] to come with new human tracking algorithm that is robust to background and lighting changes and does not require special hardware components. In addition this system can handle various scales of human images. The proposed system uses sum of absolute difference (SAD) with thresholding as has been described in [2] and compares the output with the predefined person pattern using the technique which has been described in [1]. Using the combination between [1,2] approaches will enhance the performance and speed of the tracking system since pattern matching has been performed according to just one pattern. After matching stage, a specific file is created for each tracked person, this file includes image sequences for that person. The proposed system handles shadows removal, lighting changes, and background changes with infinite pattern scales using standard personal computer.

1. Introduction

Human tracking system plays a critical role in many applications such as surveillance and robot applications. Human tracking systems can be divided into two primary types: Systems without time constraints (off-line systems), and on-line systems (real-time systems), which require capabilities to handle time constraints, like tracking in robot and surveillance applications. The basic idea in human tracking is to find a moving object and compare it with a human pattern to extract human motions. The goal of this paper is to create an adaptive system for human tracking, which is flexible to handle variations in lighting, and/or background, with real-time (indoor or outdoor) performance using ordinary personal computer without the need for camera calibrations, or predefined conditions. The rest of this paper is organized as follows. In Section 2, the design and implementation of the proposed human tracking system is presented. In section 3 implementation issues are discussed, followed by a summary and future work in Section 4.

2. An Invariant Real-Time Human Tracking System

2.1. Introduction

The challenge in this paper is to create an invariant real-time human tracking system using an ordinary personal computer without any additional hardware, initial conditions, or assumptions. In addition this system can handle various scales of human images.

2.2. Description of the Proposed Algorithm

A block diagram of the proposed human tracking system is shown in Fig. 1. The proposed system is divided into two parts: invariant processing and human tracking. These two parts can be described as follows:

Invariant Processing. The color space used within our system is RGB (pixel color is represented by red, green, and blue components) because this is the common representation of color images within a computer. Since most camera types produce images using RGB color space, there is no need to convert to other color spaces. Converting from RGB to any other color space takes time which depends on image size.

Lighting Changes and Shadows Adaptation. In this part, the process is based on applying a modified power law transformation to each pixel in the captured frame; this operation compensates lighting changes and removes shadow from image sequences.

The power law transformations have the following form [3]:

$$s = cr^{\gamma} \qquad (1)$$

Where c and γ are positive constants, r is the input pixel value, and s is the output pixel value. The output from Eq.1 must be rescaled to adapt the full range of RGB color space as shown in Eq.2

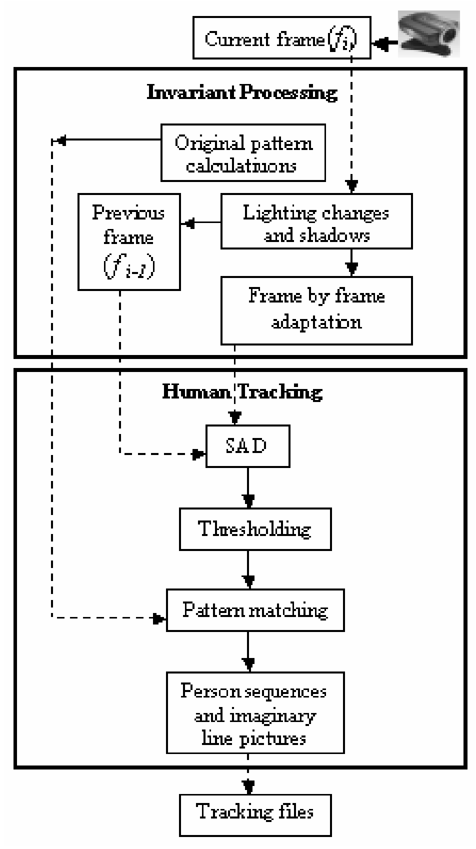

Current frame(f_i)
Invariant Processing
Original pattern calculations
Previous frame (f_{i-1})
Lighting changes and shadows
Frame by frame adaptation
Human Tracking
SAD
Thresholding
Pattern matching
Person sequences and imaginary line pictures
Tracking files

Fig. 1. System block diagram

$$op = (-1 \times Min + p) \times \frac{255}{Max} \tag{2}$$

where;

p is the input pixel value,

op is the output pixel value,

$Min = c \times 0^{\lambda}$,

$Max = c \times 255^{\lambda}$,

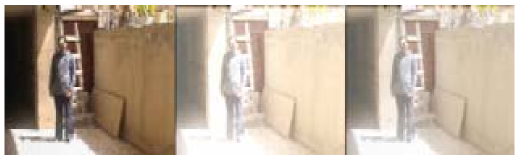
c and λ are positive constants from the power law.

The modified power law is defined as in Eq.1; however it does not require the scaling defined in Eq.2. Without using the scaling defined in Eq.2, the RGB color value will be compressed for every pixel to be closer to each other. The removal of the scaling operation is achieved by properly selecting the values of *c* and γ in Eq.1. Presented work in [2] shows that the best value for γ must be 0.32. This value has been selected by combined subjective and objective evaluation. In addition, instead of the scaling process, the power law with constant *c* greater than one has been used. Using this technique, the time needed to perform scaling process has been saved and enhances real-time performance.

Shadow is a semi-transparent region in the image, which retains a (reduced contrast) representation of the underlying surface pattern, texture or RGB color value. Applying the modified power law will decrease or eliminate shadows appearance from each frame according to shadow strength. In addition, it improves captured image appearance even in dark regions.

The adjusted formula for the power law transformation allows working with variety camera types without predefined calibration.

Comparing our modified power law with the typical power law when γ = 0.32, we found that our modified power law gives better result for shadows removal as shown in Fig.2. Fig.2-a is the captured image, Fig.2-b is the image after applying the modified power law, and Fig.2-c is the image after applying the typical power law.



(a) (b) (c)

Fig. 2. Power law comparison when γ = 0.32

Original Pattern Calculations. Since the presented work deals with human tracking, a good person model is introduced to extract only human motions. We have designed our pattern shown in Fig.3 to deal with all human motion situations (frontal or profile motions), where in the worst case we have *50%* matching between the pattern and the tracked person. We have applied the idea in [1] to overcome scaling problem and its related delay, without the need of pattern codebook and the scaling process shown in [2]. This idea is concerned with problems of construction of specialized computing device for image processing. This device is used for classification of human body shapes in conditions of various image scales of the same object using our pattern.

In our algorithm, the basic frame size is QCIF, which represents a standard frame size for camera because larger frame will not gain any additional enhancement in the tracking process since we don't need details about the tracked person face. Larger frame size can be adapted without any shortcoming.

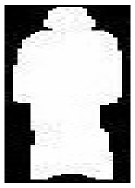


Fig. 3. The pattern used in our system

We use class indicator with different scales of various objects described by the formula below: parameters perimeter, it is known that the ratio:

$$\rho = P / \sqrt{S} \tag{3}$$

Where: **P:** perimeter and ***S*:** area of images objects.

Note that *P* and *S* are invariant to the changes of the scale of any polygon. The correctness of Eq.3 is shown in [1]. Also, we see that the ratio (3) is invariant to the change of square scale. Now let us show the invariant of the ratio (3) to the scale changes for any shape.

In Fig.4, it is shown that any shape may be shown as a sum of squares.

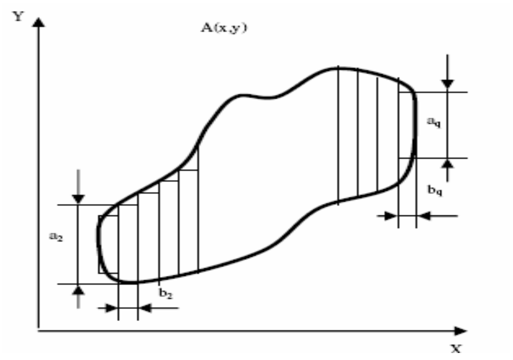


Fig. 4. Shape A as a sum of squares

For shape A(x,y), the ratio (3) may be given in the following equation:

$$\rho_A = \frac{P_A}{\sqrt{S_A^{area}}} = \frac{a_1 + a_q + 2\sum_{i=1}^{q} b_i}{\sqrt{\sum_{i=1}^{q} a_i b_i}} \qquad (4)$$

Where; a1,aq are the length of the first and last squares; bi(i=1,q) are the width of squares. When changing the scale of the image A(x,y) in k times, we have:

$$\rho_k = \frac{P_k}{\sqrt{S_k^{area}}} = \frac{ka_1 + ka_q + 2\sum_{i=1}^{q} kb_i}{\sqrt{\sum_{i=1}^{q} k^2 a_i b_i}} = \frac{a_1 + a_q + 2\sum_{i=1}^{q} b_i}{\sqrt{\sum_{i=1}^{q} a_i b_i}} = \rho \qquad (5)$$

Where; Pk and *Sk* are correspondingly perimeter and area of the figure, the scale of which is changed in k times.

To calculate the perimeter of any k image, we consider that it consists of the length of separate pixels. As shown in Fig.5, any pixel of the image may have one of the following numbers of adjacent pixels.

Pixels marked on Fig.5 by the digit 1, have 4 neighboring pixels (up, down, right, left), the contribution of theses pixels to the length of perimeter is equal zero.

Pixels marked on Fig.5 by the digit 2, each have three neighboring pixels; their contribution to the perimeter length is equal to the length of one pixel 1d.

Pixels marked on Fig.5 by the digit 3, they have two neighboring pixels; their contribution to the perimeter length is equal to 2d.

Pixels marked on Fig.5 by the digit 4 have only one neighboring pixel; their contribution to the perimeter length is equal to 3d.

Pixels marked on Fig.5 by the digit 5; their contribution to the perimeter length is equal to 4d.

By analyzing Fig.5 and the given reasoning, we can calculate the length of perimeter of any image by the following equation:

$$P=N\ 1\ d + 2N\ 2\ d + 3N\ 3\ d + 4N\ 0\ d \qquad (6)$$

Where; $N\ r = 1,3$ is the number of excited neurons which have r neighboring excited pixels in the image;

N 0 is the number of neurons which don't have neighboring pixels in their space.

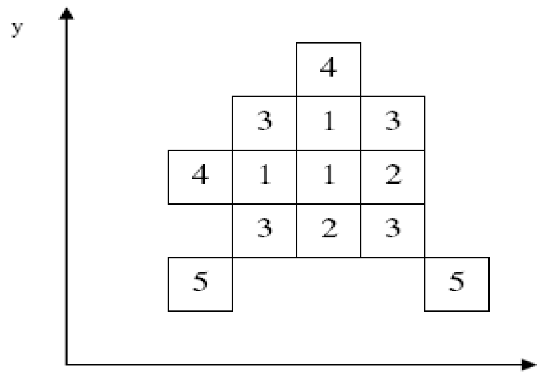


Fig. 5. Contribution to of different Pixels to the perimeter length of image

Frame by Frame Adaptation. The final invariant issue is background manipulation. Background estimation using training period or statistical methods will take time in the process of generating a reference background. In addition, that reference background must be re-estimated in case of background changes. To overcome background changes problem, we do not have any background reference to be created or updated. Our algorithm uses frame by frame method to extract changes between two image frames f(x,y,ti) and f(x,y,tj) taken at times ti and tj, respectively by comparing these two frames pixel by pixel. Using this method, we can adapt to different backgrounds. In addition, we save time and increase robustness against light changes which make this method suitable for real-time implementations.

Human Tracking. In this part, we present the design of a human tracking algorithm that is compatible with the presented invariant image processing algorithm. This compatibility makes the algorithm more robust in outdoor and indoor with lighting changes and without any background initialization, or camera calibration. In tracking process, comparing frame sequences with each other including moving objects, results in difference of two images canceling the stationary elements, leaving only non-stationary image components. These components are the main element in the human tracking process.

Sum of Absolute Difference (SAD). In our algorithm after performing invariant processing, a difference image between two images taken at time ti and tj is defined using sum of absolute difference (SAD) for each RGB color channel. Although the sum of squared difference (SSD) based methods are widely used as the core component of many tracking algorithms and have been already integrated in many systems. We have decided to use SAD to calculate frame differences and perform matching process instead of SSD.

From simulations we have found that, SAD used in (Fig. 6-a) eliminate noise effect, whereas using SSD in (Fig.6-b) increases noise and reducing the system speed.

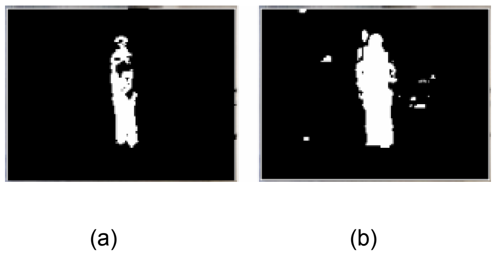


Fig. 6. Comparison between SAD and SSD

We select a block size of 3×3 *pixels* to perform SAD as shown in Fig. 7, because this size is the smallest size can be taken to enhance operation by neighborhood information and to make the result smooth. Larger block size will increase the consumed time to perform the operation without large enhancement; this selection has been done according to subjective evaluation. For a block of size *3×3*, SAD is defined for the three color components (R, G, B) as shown in Fig.7.

```
for i=1 to frame width-1
  for j=1 to frame height-1
      $SAD_R = \frac{1}{3\times 3}\sum_i^3\sum_j^3 \left| f_R(t_i) - f_R(t_j) \right|$
      $SAD_G = \frac{1}{3\times 3}\sum_i^3\sum_j^3 \left| f_G(t_i) - f_G(t_j) \right|$
      $SAD_B = \frac{1}{3\times 3}\sum_i^3\sum_j^3 \left| f_B(t_i) - f_B(t_j) \right|$
  next j
next i
```

Fig. 7. Pseudo code for SAD calculation

Thresholding. The resulting difference image from the SAD step is used to extract moving objects in contiguous frames. Thresholding of this difference image yields a binary image containing “foreground” regions where movement was detected. Threshold value, *T*, was selected to be 50 using subjective evaluation after various studies and experiments to lighting and shadow properties. Figure 8 shows the Pseudo code of the thresholding process.

At the end of this step the output frame represents moving objects in white and background objects in black. Here, we called the resulting image binary image.

Pattern Matching. Recognizing human within a frame according to a person model using traditional cross correlation is time consuming so it can not accommodate real-time restriction. To overcome this problem, some implementations use fast cross correlation which performs matching faster than the traditional cross correlation. But even fast cross correlation still consumes time to deal with small illumination effects. Since in our algorithm we have compensated illumination effects by using our invariant algorithm, we didn't use cross correlation techniques to save time. So Eq.3 is used instead of cross correlation. We apply Eq.3 using our pattern (shown previously in Fig.3) and compare the result with the output of the thresholding process, this increases speed and save time in performing matching to the entire frame using SAD.

```
for i=1 to frame width-1
  for j=1 to frame height-1

    if $SAD_R$(i, j+1)OR $SAD_G$(i, j+1)OR $SAD_B$(i, j+1)> T then
     if $SAD_R$(i, j-1)OR $SAD_G$(i, j-1)OR $SAD_B$(i, j-1)> T then
       if $SAD_R$(i+1, j)OR $SAD_G$(i+1, j)OR $SAD_B$(i+1, j)> T then
         if $SAD_R$(i-1, j)OR $SAD_G$(i-1, j)OR $SAD_B$(i-1, j)> T then
          image(i, j)= 255 //white
    else
     image(i, j)= 0 //black
         end if
       end if
     end if
    end if

  next j
next i
```

Fig. 8. Pseudo code of thresholding process

Person Sequences and Imaginary Line Pictures. After identifying region of human motion, a special file is created containing all paths for every tracked person. Our system deals with occlusion, simply by extracting motion direction and clothes color for each tracked person from the file. Within the tracking process, if the person enters predefined imaginary line (to make sure that the person stands in good position to take his face detail) the system will capture a picture for that person, which can be used in any face recognition technique. We define the starting of the imaginary line at (0, frame height/4) and the end of the line at (frame width, frame height/4) to obtain clear picture for each tracked person.

3. Implementation Issues

In this section, we discuss the implementation issues. Our goal is to create a robust, adaptive real-time human tracking system that is flexible enough to handle variations in lighting and/or background without using special hardware or predefined conditions.

To achieve this goal we have implemented our system with two camera types: Panasonic NV-DS65 and Kinstone PC CAMERA.

In addition, to prove that our system adapts to a variety of camera types, our algorithm has been applied to images taken from various web sites and captured by different camera types. Our system is implemented using a personal computer Pentium(R) 4 with the following specifications: CPU speed 1.70GHz with 256 MB of RAM.

Our system has been implemented using visual basic 6 (VB6). To increase our system speed in VB6, we use multithreading with two threads, one for the invariant processing and the other for the tracking process. Using multithreading with two threads in our implementation increases the speed to 50 *ms* per frame. Increasing the number of threads beyond two threads results in a higher processing time per frame, this is due to the fact that more time is required to manage the threads.

We note that in the case of human tracking real-time, performance will be more than established using three frames per second as stated in [4]. This is because in the case of human motion no large displacements occur within a time that is less than half second. In our system each captured frame is of size 176×144 pixels. This size is selected to help in noise elimination and increase the system speed. On the other hand frame of size 320×240 pixels can be used with speed of 65 *ms* per frame. It can be seen that our system speed can be improved using faster personal computer (more than 1.70GHz).

3.1. Experimental Results

Our experimental results are divided into three parts: lighting changes experiment, shadows removal experiment, and tracking experiment. Details for each experiment are now presented.

Lighting Changes Experiment. For lighting changes experiments, two captured frames were selected for each experiment. The two views were images under different lighting conditions. IRHTS is applied for each two views. The resulting difference image (binary image) represents the accuracy for lighting changes adaptation, which must be totally black image for full adaptation.

In Fig.9 images (a) and (b) represent scene with varying illumination. Images (c) and (d) represent the output images after applying the invariant part of the IRHTS for the two images (a) and (b) respectively. Image-e represents the SAD between (c) and (d).

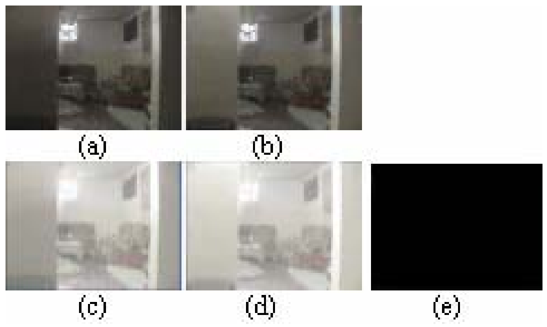


Fig. 9. Lighting changes case

Shadow Removal Experiment. Shadow removal experiments have been applied for images captured using our two camera and images captured using different camera types from various web sites and papers. The results for images that were taken using our two cameras are shown in Fig.10.

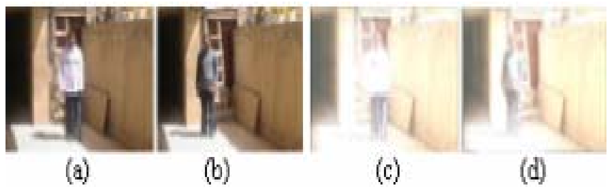


Fig. 10. Shadows removal

The results for images that were taken using camera from various web sites and papers are shown in Figures (11 and 12). where image-a represents the captured image, and image-b is the output image from the IRHTS invariant algorithm.

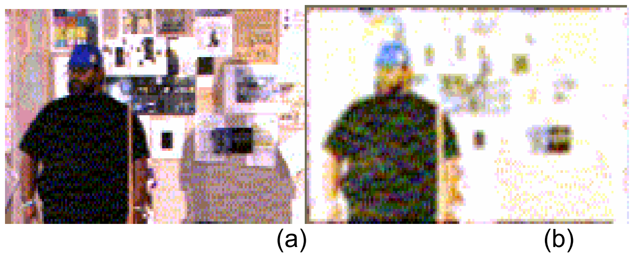


Fig. 11. Shadows removal (unknown camera1)

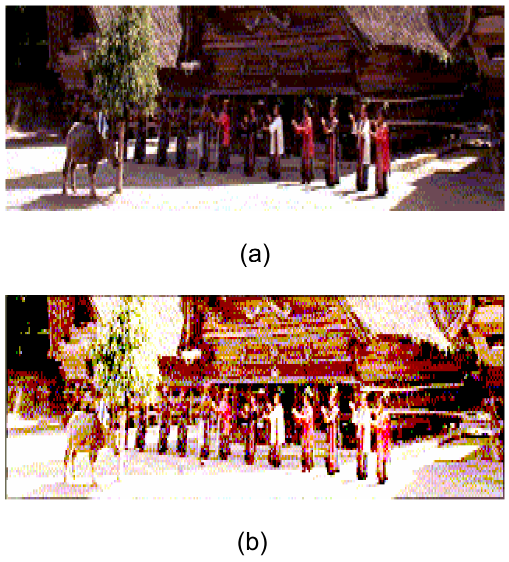


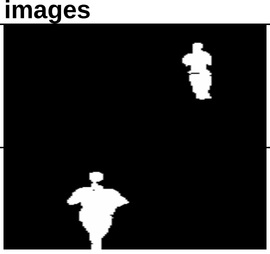
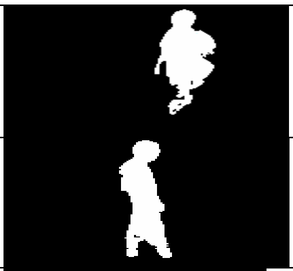
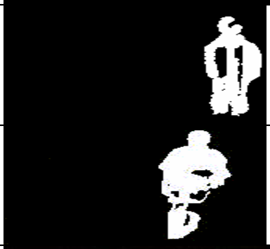
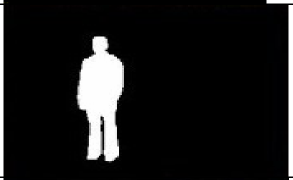
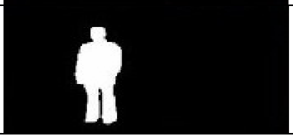
Fig. 12. Shadows removal (unknown camera2)

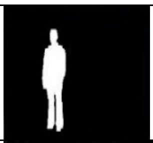
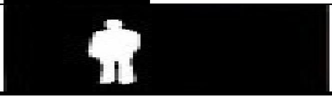
Form the figures shown, it can be seen that applying the IRHTS eliminates large amount of shadows. If small areas of shadows still appear, the IRHTS removes them in the SAD part of the algorithm.

Multi Scales Experiment. For multi scales experiment, different sequences for human with different sizes were selected and compared with our pattern using Eq.3. The output of this experiment is shown in table 1, which contain each image and its similarity percentage according to our pattern.

Tracking Experiment. For tracking experiments different image sequences in different environments and conditions were selected. The result of tracking process is shown as a binary image (black and white). Black area represents background and stationary objects; white regions represent the tracking person. In each tracking sequence, a picture is attached which is captured for the tracked person when that person enters the predefined imaginary line.

Table 1. : Pattern similarity percentage

images	percentage
	96.7
	94.4
	90
	92
	90
	91
	99.6
	98

	97.4
	96

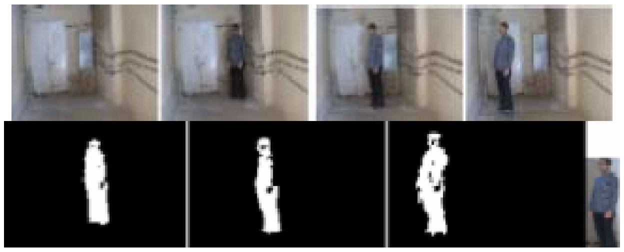


Fig. 13. Indoor

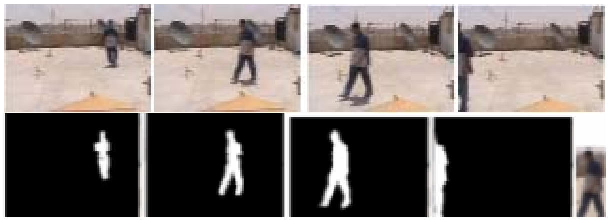


Fig. 14. Outdoor

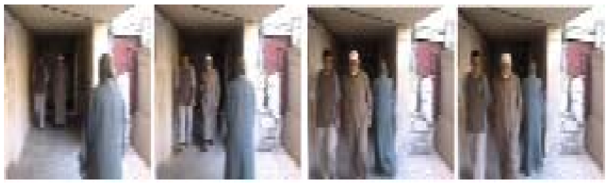


Fig. 15. More than one person sequences

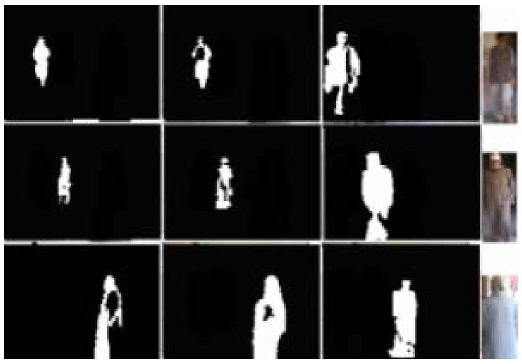
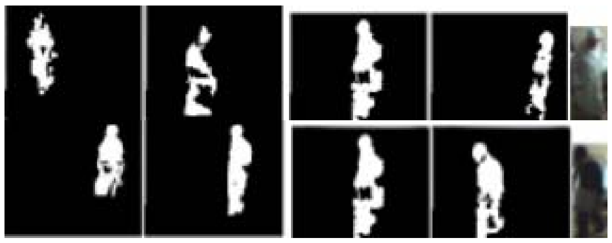


Fig.16. More than one person

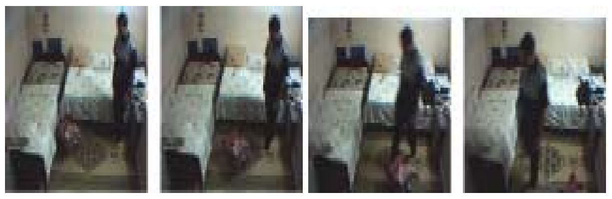
More than one person with occlusion sequences



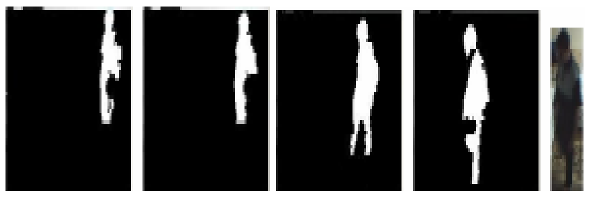
More than one person with occlusion sequences tracking result

Fig.17. More than one person with occlusion

The image sequences that were taken for tracking with moving object (to show that IRHTS just tracks human) are shown in the next figure.



Tracking with moving object sequences



Tracking with moving object sequences result

Fig.18. Tracking with moving objects sequences

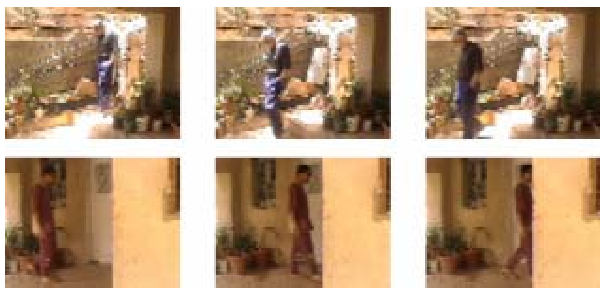
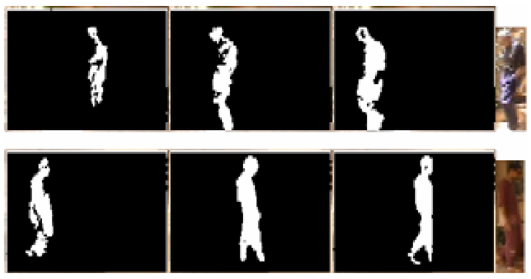
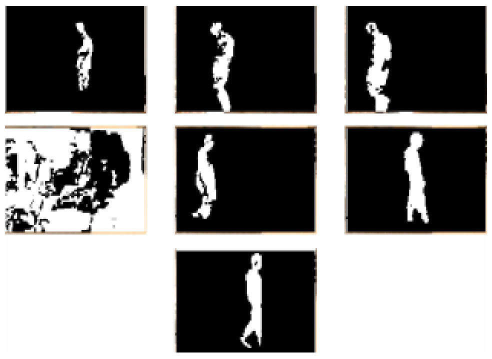


Fig.19. Background changes sequences

The image sequences that were taken with different background for the same scene are shown in Fig.19 (first row). The result after applying the IRHTS for these image sequences are shown in Fig.20 (first row). As shown in Fig. 20 (second row), changing background will cause error in one frame, which will be ignored in tracking result files (Fig. 20 (first row)).



Background changes sequences tracking result



Background changes sequences thresholding result

Fig.20. Background changes sequences

4. Conclusion and Future Works

Since the two most important parameters in tracking systems are lighting and background changes, the work in this paper present the design and implementation of an invariant human tracking system that is robust to background and lighting changes in addition to shadows removal and does not require special hardware components or camera calibrations. Our system can handle the problem of different scales patterns and frames using Eq.(3) which increases system speed and accuracy to handle various human sizes using one predefined pattern.

In our paper, the work can be divided into two steps: the first step is dealing with invariant conditions. The second step is combining invariant algorithm from the first step with the tracking system to perform tracking in real-time.

Our system deals with occlusion simply by extracting motion direction for each tracked person combined with that person clothes color from the file. In the tracking process if a person enters a predefined imaginary line, the system will capture a picture for that person, which can be used in any face recognition technique, to identify the person.

The proposed system is robust and does not require the information of scene geometry, camera parameter, and lighting condition at all. As a future work and to enhance the system, we suggest studying our system with moving camera and a pan-tilt mounted zoom camera.

References

1. Rashad J. Rasras, "Neurocomputing Device for Image Processing and Classification", Journal of Computer Science (Special Issue): 89-91, 2005.
2. Mazen Abu_zaher, "An Invariant Real-Time Human Tracking System", Master thesis (supervised by Dr.Fayez Idris), Faculty of Graduate Students, Jordan University of Science and Technology: 2006.
3. Gonzalez C., and Woods E., Digital Image Processing, Prentice Hall, second edition, 2002.
4. Siebel N. and Maybank S., "Real-Time Tracking of Pedestrians and Vehicles", In the Second IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, 2001.

Dr. Fayez Idris the Director of the Center of Information Systems and Technology at the German-Jordanian University (GJU), Jordan. Dr. Idris received his PhD and MSc in Computer Engineering from the University of Ottawa, Canada in 2000 and 1993, respectively. His research interests include image processing, Arabic text processing, and network coding.

Rashad J. Rasras is assistant professor at Faculty of Engineering Technology –Albalqa' Applied University (http://www.fet.edu.jo). He is also currently head of computer engineering department, and member of Jordan Engineering Association.. His research focus is on image processing and neural networks, modeling complex process and artificial intelligence. He is experienced in several development environments, coding mostly C++, Assembly , and Oracle Data base. He has many publications in fields of neural networks, neural networks applications, and image processing.

Ibrahiem M. M. El Emary is an assistant professor of computer engineering and a researcher in computer sciences and engineering, currently working head of computer engineering department at the Faculty of Engineering, Al Ahliyya Amman University (http://www.ammanu.edu). He is experienced in teaching various computer engineering courses like: computer networks, programming with C++, Database Design, Database Management Systems, Artificial Intelligence, Expert Systems, Digital Systems, Logic Design, Software Engineering and Automatic Control. He published more than 60 papers focusing on: application of Intelligent techniques in managing telecommunication systems, Analyzing various types of simulation techniques, Using AI in pattern recognition, Application of AI in controlling electric power systems, Designing software Agents for handling the various jobs of telecommunication networks, and developing models for analyzing and processing color images.

Mazen Abu_Zaher is now lecturer at Faculty of Engineering Technology – Albalqa' Applied University. He got his B.S. in computer engineering at Faculty of Engineering Technology –Albalqa' Applied University in 2004, and M.Sc in computer engineering at Jordan University of Science And Technology in 2006. He is interested with the following languages: ASP.NET, HTML, Java Script, C++, Visual Basic, SQL, Assembly 8086 &8085. His current research focuses on, and interested in VLSI, computer architecture, and robot applications.