IEICE TRANS. INF. & SYST., VOL.E86–D, NO.12 DECEMBER 2003

PAPER

Moving Target Detection and Tracking Using Edge Features Detection and Matching

Alireza BEHRAD† *and* **Seyed AHMAD MOTAMEDI**†, ***Nonmembers***

SUMMARY A new algorithm for fast detection and tracking of moving targets using a mobile video camera is presented. Our algorithm is based on image feature detection and matching. To detect features, we used edge points and their accumulated curvature. When the features are detected they are matched with their corresponding points using a new method called fuzzy-edge based feature matching. The proposed algorithm has two modes: detection and tracking. In the detection mode, background motion is estimated and compensated using an affine transformation. The resultant motion-rectified image is used for detection of the target location using split and merge algorithm. We also checked other features for precise detection of the target. When the target is identified, algorithm switches to the tracking mode, which also has two phases. In the first phase, the algorithm tracks the target with the intention to recover the target bounding-box more precisely and when the target bounding-box is determined precisely, the second phase of tracking algorithm starts to track the specified target more accurately. The algorithm has good performance in the environment with noise and illumination change.

key words: *moving target detection and tracking, feature detection, feature correspondence, edge matching*

1. Introduction

Visual detection and tracking is one of the most challenging issues in computer vision. Applications of the visual detection and tracking are numerous and they span a wide range of applications including surveillance system, vehicle tracking and aerospace application, to name a few. There are many difficulties with moving target detection and tracking algorithms such as noise of the target, change in scene illumination, change in target geometry, real-time constraint, camera motion and occlusion, to mention a few. Detection and tracking of abstract targets (e.g. vehicles in general) is a very complex problem and demands sophisticated solutions using conventional pattern recognition and motion estimation methods. Motion-based segmentation is one of the powerful tools for detection and tracking of moving targets. For example difference-based methods [1] employ the difference of two consecutive frames to extract moving parts of the scene. However, the conventional difference-based methods fail to detect moving targets when the camera is also moving. In the case of mobile camera all of the objects in the scene have an apparent motion, which is related to the camera motion so other considerations and restrictions are required. For example in [2], it is assumed that the target constitutes small portion of the scenery and in [3], it is assumed that the camera motion is only because of pan and tilt motions and using an active camera these motions are compensated. Optical flow techniques [4], [5] are other well-known techniques for motion analysis, which calculate velocity vectors for all pixels in the image. The main difficulty with these techniques is their high computational overhead as well as their lower efficiency in the presence of noise and illumination changes. Most of these algorithms also suffer from aperture problem. Active contour models [6], [7], also called snakes, are used for contour extraction as well as contour tracking. In these methods the tracking process is performed by minimizing a predefined energy calculated in the number of discrete points from the contour of the target, called snaxel. These techniques have lower computation overhead and are robust against smooth illumination change of the scene, but they are very sensitive to the target noise and occlusion.

Feature based methods [8]–[12] are other well-known methods for target tracking, which could resolve many tracking problems such as camera motion, real-time constraint and geometry change of the target. In these methods a few discrete features such as corners, edges or lines are extracted and by matching these features with similar features in the next frames, motion vectors or motion model between two consecutive frames are calculated. This technique has two main advantages; first since the motion vectors are calculated for small number of points, the computation overhead of the algorithm is low so the algorithm is suitable for real-time applications, and second, because a feature is an image property, which image motion can be measured more accurately, therefore, the overall accuracy of the algorithm is improved.

Various methods have been proposed for feature extraction such as edge features [13], lines [14], corners and also regions. Among these methods corners are more popular and robust and variety of algorithms are proposed to extract corners such as Moravec Operator [15], Plessey algorithm [16], SUSAN corner detector [17] and Mokhtarian CSS method [18], to name a few. There are also different methods of feature matching such as sum square difference(SSD), normalized correlation and using corners properties [19].

Manuscript received March 14, 2003.
Manuscript revised July 13, 2003.

†The authors are with the Electrical Engineering Department, Amirkabir University of Technology, Tehran, Iran.

Existing techniques for tracking a set of discrete features generally fall into two categories [12]: two-frame based and long-sequence based. In two-frame based methods [8], [11] the process of finding corresponding points over an image sequence is broken into successive, yet independent problems of two-view matching. Long-sequence based methods [9], [10] in other side employ smoothness constraints to exploit temporal information existing in the sequence. In essence, when the motion of the camera and target is smooth such that smoothness constraints hold, long-sequence based methods are likely to outperform two-frame based methods. On the contrary, if the movement of the target or camera results in non-smooth image motion, two-frame based methods give good results.

In this paper we propose a new method for detection and tracking of moving targets using a mobile monocular camera. We have devised feature-based algorithms for non-uniform motions, however the designed tracking algorithms are capable of extracting non-uniformity in the motion and using the existing uniformity of the motion for fast and precise tracking purpose. In order to detect and track the targets more accurately, different algorithms for detection and tracking purposes have been utilized. At first, the algorithm is in the detection mode and when a target is found, the algorithm switches to tracking mode, which also has two phases. In tracking algorithms we benefit from the advantages both long-sequence and two-frame based methods. To detect image features and match them with their corresponding points, new methods have been utilized, which are more robust than previous methods. We have used edge points and their accumulated curvature for feature detection and the matching is carried out using a new method called fuzzy-edge based feature matching. These methods also have good results in the case of noisy image sequences or images with illumination change.

This paper is organized as follows. In the next section the proposed algorithms for feature extraction and matching are described. In Sect. 3, the detection procedure is discussed. Section 4 describes the tracking algorithm. Experimental results are shown in Sect. 5 and conclusions appear in Sect. 6.

2. Feature Extraction and Correspondence

As mentioned before, noise and change in illumination of the images are some of the factors may deteriorate the performance of moving target detection and tracking algorithms. In order to deal with these difficulties, it is necessary that both feature detection and feature matching algorithms have lower sensitivity to the presence of noise and illumination change. Edge points are proper features of image, which have lower sensitivity to the image noise and illumination variance. However when the edge points are in the form of straight lines or lines with lower curvature, the corresponding points can not be determined precisely, which is called aperture problem. To solve aperture problem and determine the precise position of corresponding points, locations should be selected as edge features, which have enough information for matching process. The CSS corner detector is a suitable corner detector, which extracts the corners of image from the contours of edge-detected image. Although the CSS corner detector considers the edge junctions and edge curvature, which are good features for edge matching, the curvature of only one contour is considered.

To detect more appropriate edge features, we have developed an edge feature detector algorithm, which considers the accumulated curvature of edge pixels in the match window. The algorithm also considers the number of edge pixels in the match window, which is another useful factor for correct edge matching. Our algorithm consists of following steps:

1. Extract the edge contours from the input image using any good edge detector such as Canny.
2. Fill small gaps in edge contours. When the gap forms a T-junction, mark it as a T-corner.
3. Calculate the curvature of Gaussian smoothed edge pixels. In our algorithm it is not necessary to calculate the curvature at different scales. To calculate the curvature for edge pixels, edge contours are represented as parametric vector $r(u) = (x(u), y(u))$. Then the curvature is calculated using the following equation [18]:

$$\kappa(u,\sigma)=\frac{X_u(u,\sigma)Y_{uu}(u,\sigma)-X_{uu}(u,\sigma)Y_u(u,\sigma)}{((X_u(u,\sigma)^2+Y_u(u,\sigma)^2)^{3/2}} \tag{1}$$

4. Find the minimum and average of the absolute curvature values in the image and denote them as κ_{min} and κ_{ave}.
5. Assign the value of 0 to non-edge pixels and the value of $|\kappa(u,\sigma)| + \alpha\kappa_{ave} + (1-\alpha)\kappa_{min}$ to edge pixels. Where α is a small positive number (typically $\alpha = 0.1$) and the constant term $\alpha\kappa_{ave} + (1-\alpha)\kappa_{min}$, which is added to all edge pixels, aims to consider the density of edge pixels in the detection of proper edge features. Here $|\kappa(u,\sigma)|$ denotes absolute value of $\kappa(u,\sigma)$,
6. Scan the image and calculate the accumulated curvature for each pixel. The accumulated curvature is the sum of assigned values in the previous step and summation is carried out in the small windows centered on pixels,
7. Find the local maximums of accumulated curvature values and apply a threshold to suppress weak features.

2.1 Fuzzy-Edge Based Feature Matching

When the feature points are detected, the second step is to find the corresponding points of features in the second image. In addition to noise and illumination change of the scene, change in geometry and size of objects in the two images being matched are other factors, which might deteriorate the efficiency of the matching process. In moving target detection and tracking applications especially when the camera is also mobile, the three-dimensional movement of the target and camera or zoom operation of the camera may result in geometry change of the target. Therefore the direct matching of edge pixels, such as edge correlation is not a proper method to find corresponding points in these applications. To alleviate this problem, a new edge matching method is utilized which is called fuzzy-edge based feature matching. In this method edge pixels are described as fuzzy values, which are called fuzzy-edges. To describe edge points as fuzzy values we assign the value of 1 to edge points and a value between 0 and 1 to the points that are in the neighborhood of the edge points. This value is determined based on the distance from the edge points and the fuzzy membership function, which is used for this purpose. Different membership functions [20] such as triangular and trapezoid can be used for the construction of fuzzy-edges. Since a point may take different fuzzy values from its neighborhoods, the maximum of assigned fuzzy values is considered as a final value. Figure 1 shows a typical edge-detected image and related fuzzy-edges using triangular membership function.

When the fuzzy-edges are obtained for both images being matched, SSD or normalized correlation method is applied to fuzzy-edges to locate the corresponding points of features in the second image.

2.2 Subpixel Matching

The corresponding points which are obtained using similarity measure of fuzzy values, have pixel resolution.

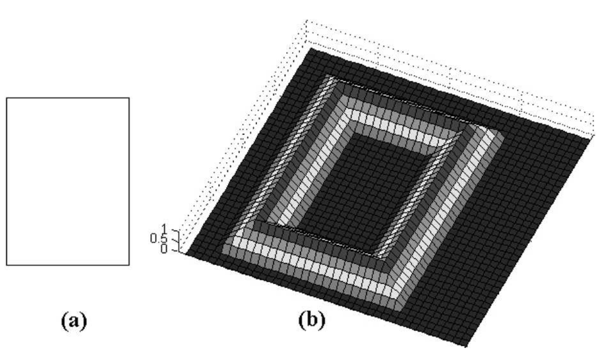


Fig. 1 (a) An edge-detected image (b) fuzzy-edges using triangular membership function with the base of 7.

In some of applications it is also required to calculate the corresponding points with sub-pixel accuracy. Since our method is a kind of area-based matching, it is possible to calculate the sub-pixel corresponding points using parabola fitting. In our implementation we used least square parabola fitting on similarity profile in the neighborhood of the optimum value to identify the sub-pixel peak location and calculate the sub-pixel corresponding point. To reduce the computational overhead of the fitting process, two one-dimensional parabola fitting for x and y directions are used.

2.3 Increasing the Speed of Matching Algorithm

In the case large disparities, the search space for finding match points is large and this makes the computation overhead of the algorithm largely increase. Multiresolution feature matching is one of the mostly used methods to overcome the problem of large disparities. In multiresolution matching, different levels of resolution of the image are created and the match is usually applied in the highest level and then propagated to lower levels. Since the disparities become small at high levels, this method considerably reduces the computational overhead of matching algorithm.

Our fuzzy-edge based matching algorithm can also benefit from using multiresolution method. Our algorithm for multiresolution feature matching has following steps:

1. Build edge pyramid of images. The algorithm to construct edge pyramid is explained bellow.
2. Perform matching at lowest resolution.
3. Propagate matches down to next higher resolution.
4. Correct match estimates and continue to next level.

Suppose the edge detected image g_0 is represented initially by the array of edge contours:

$$r_{0n}(u) = (x_{0n}(u), y_{0n}(u)) \qquad n = 1:N \tag{2}$$

where N is the number of edge contours in the image. Also suppose that the image contains C columns and R rows of pixels. This image becomes the bottom or zero level of the edge pyramid. Pyramid level 1 image g_1, which contains $C/2$ columns and $R/2$ rows of pixels, is represented by:

$$r_{1n}(u) = (x_{1n}(u), y_{1n}(u)) \qquad n = 1:N \tag{3}$$

where $r_{1n}(u)$ are reduced and low-pass filtered versions of $r_{0n}(u)$ and are calculated as follows:

$$X_{0n}(u,\sigma) = x_{0n}(u) \otimes g(u,\sigma) \tag{4}$$
$$Y_{0n}(u,\sigma) = y_{0n}(u) \otimes g(u,\sigma) \tag{5}$$
$$R_{0n}(u,\sigma) = (X_{0n}(u,\sigma), Y_{0n}(u,\sigma)) \tag{6}$$
$$r_{1n}(u,\sigma) = R_{0n}(2u,\sigma)/2 \tag{7}$$

Fig. 2 Different levels of edge pyramid for a typical image (a) original image (b) edge image level 1 (c) edge image level 2 (d) edge image level 3.

where $g(u, \sigma)$ denotes a Gaussian function of width σ and $\otimes$ represents one-dimensional convolution.

The pyramid level 2 image, is then obtained from image level 1 by applying the same algorithm described above. The algorithm is repeated to build all levels of edge pyramid. Figure 2 illustrates the contents of an edge pyramid generated with $\sigma = 2$.

3. Moving Target Detection

It is assumed that the camera is mobile, therefore the first stage to detect true moving target is to cancel the camera motion or in other word the apparent background motion. Several methods have been proposed to describe the motion of an object between two frames. Among these methods three transformations: affine, projective and polynomial, are most commonly used in target detection and image registration. The affine transformation accounts for rotation, translation, scaling and skew between two images as follows:

$$\begin{pmatrix} X_i \\ Y_i \end{pmatrix} = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix} + \begin{pmatrix} a_5 \\ a_6 \end{pmatrix} \tag{8}$$

where (x_i, y_i) are coordinates of points in the current frame and (X_i, Y_i) are coordinates of points in the next frame and $a_1 - a_6$ are motion parameters. Although projective and polynomial consider more distortions, affine transformation is easy to use and is proper for real-time applications. Furthermore, in this work the motion model is computed for the background in two consecutive frames so various distortions are expected to be small. Therefore affine transformation has been selected. This transformation has six parameters; therefore, three matching pairs are required to fully recover the motion. However it is necessary to select the three points from the background to assure an accurate model for camera motion.

To estimate camera motion it is assumed that the moving targets constitute small region (less than 50%) of the scene, therefore a motion model that describes the dominant motion of the scene is the motion model of the background and can be calculated using methods such as Least Median Square(LMedS) statistic [21], [22] or Random Sample Consensus Paradigm(RANSAC) [23] as follows:

1. Select M random sets of three feature points: (x_i, y_i, X_i, Y_i) for i=1,2,3, where (x_i, y_i) are coordinates of the feature points in the previous frame, and (X_i, Y_i) are their matches in current frame. M is the minimum number of data points to make a robust parameter estimate.
2. For each data set calculate affine parameters.
3. (a) In the case of the LMedS estimator calculate median residual over all data.
   (b) In the case of RANSAC estimator calculate the size of data set consistent with estimated affine parameters.
4. Select the best solution, i.e. that with the lowest median residual or the biggest data set.

In [22] it is shown that the LMedS algorithm marginally outperform RANSAC algorithm so LMedS algorithm is used.

3.1 Extracting Target Region

When the affine parameters are estimated, they can be used for cancellation of the apparent background motion by transformation of previous frame, which is called image warping in machine vision. The position of the pixel transformed does not in general fit the discrete grid of the image, therefore brightness interpolation is necessary for image warping. We used bi-cubic interpolation which preserves fine details in the image very well and doesn't suffer from step-like boundary problem or blurring. Now the difference of current frame and transformed previous frame reveals true moving targets. Then a threshold is applied to produce a binary image. The results of the transformation and segmentation are shown in Fig. 3. As it is shown in this figure, some parts are segmented as moving targets due to noise. Increasing threshold decreases the noise effect but it also reduces the number of true detected moving points. To cope with this difficulty, two-level thresholding is used. At the first stage the algorithm uses a high threshold and any pixel that has a value greater than high threshold is presumed to be a moving pixel. Then a connection check filter is applied to remove the moving pixels with connectivity number of zero. Now any pixels that are in the neighborhood of the moving pixels and have a value greater than low threshold are also selected as moving points and this process is done recursively to extract all moving pixels.

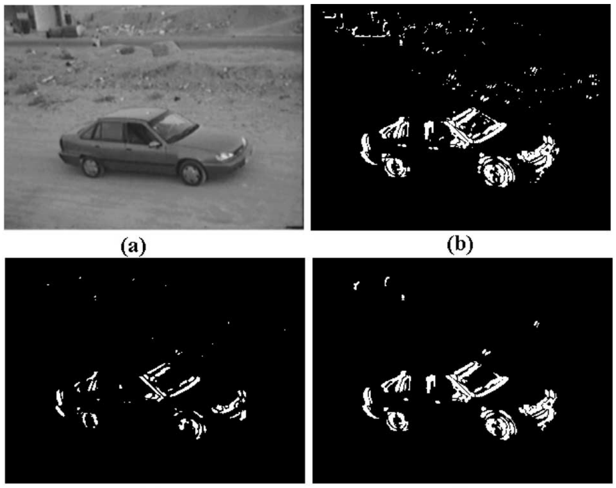


Fig. 3 The result of thresholding; (a) current frame (b) binary image using fix threshold=20 (c) binary image using fix threshold=40 (d) binary image using two-level thresholding. The calculated affine parameters are: $a_1 = 0.9923$, $a_2 = -0.004$, $a_3 = 0.0021$, $a_4 = 1.001$, $a_5 = 0.11$, $a_6 = -3.966$.

To find moving target bounding-box a split and merge algorithm is used. We scan the binary image and calculate the number of moving pixels in the windows (typically 20*20) centered on different moving pixels of binary image. Then the window with the maximum number of moving pixels is selected. If the number is larger than a threshold, split and merge algorithm is used to find target bounding-box around the selected window otherwise the search algorithm finishes. If no target is found, then it means either there is no moving target in the scene, or the relative motion of the target is too small to be detected. In the latter case, it is possible to detect the target by adjusting the frame rate of the camera. The algorithm accomplishes this automatically by analyzing the proceeding frames until a major difference is detected.

3.2 Target Verification

To make the detection algorithm robust against situations such as complex 3D camera motion and targets with larger area, a voting method is designed to verify the targets based on a-priori knowledge of the targets. Our special interest is detection and tracking of the moving vehicles. So the aspect ratio and horizontal and vertical lines are used as constraints to verify vehicles. Our experiments show that comparison of the length of horizontal and vertical lines in the target region with the area of the target will give a good clue about the nature of the target. To verify a vehicle target, we calculate the number of target pixels situated in the line segments in four different directions i.e. $0°, 45°, 90°, 135°$. Then following parameters are calculated:

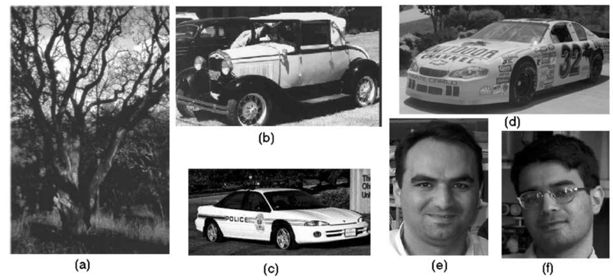


Fig. 4 The result of target verification algorithm for various targets. The calculated parameters are for $k = 0.7$ and minimum acceptable line length of 6 pixel: (a) $V = 0.067058$, $H = 0.064814$ (b) $V = 0.365194$, $H = 0.454194$ (c) $V = 0.555157$, $H = 0.572991$ (d) $V = 0.242306$, $H = 0.154140$ (e) $V = 0.049945$, $H = 0.067108$ (f) $V = 0.050315$, $H = 0.058622$.

$$V = 100 \times (N_0 + k \times (N_{135} + N_{45}))/(r \times c) \quad (9)$$

$$H = 100 \times (N_{90} + k \times (N_{135} + N_{45}))/(r \times c) \quad (10)$$

where N_0, N_{45}, N_{90}, N_{135} are the number of pixels in the line segments in $0°, 45°, 90°, 135°$ directions respectively, k is a constant number between 0 and 1 and r and c are the row and column of detected target in pixel. For vehicle targets it is checked that both V and H are greater than a threshold. In Fig. 4 some results of verification algorithm are shown. As it is depicted in this figure the values of V and H are larger for vehicle targets, therefore they can be used for target verification and increasing the robustness of detection algorithm.

4. Tracking Algorithm

After a target is verified, the algorithm switches into the tracking mode. As mentioned before, the proposed tracking algorithm contains two phases. In the first phase of the tracking algorithm, which starts after the target detection algorithm, the algorithm aims to detect the target bounding-box more precisely. For this purpose it considers motion information of the feature points and their compliance with calculated motion model. Features, which don't agree with motion model of the target during several frames, might be considered as non-target features. When the target bounding-box is determined correctly the second phase of tracking algorithm starts.

We used LMedS algorithm to calculate initial estimation of motion parameters for the target between two consecutive frames. LMedS algorithm has a good efficiency in the presence of outliers such as occlusion, however its efficiency is poor in the presence of Gaussian noise, which deeply affects the accuracy of tracking algorithms. Another difficulty with the LMedS algorithm for tracking is that it uses only the motion information of two consecutive frames for calculation of motion model. To cope with these difficulties modified

LMedS algorithm was devised, which utilizes the motion information of several frames in the case of uniform motions. The output motion parameters of the modified LMedS algorithm are then fed into a Kalman Filter or Extended Kalman Filter. The purpose of Kalman Filter is to smooth the parameters and remove Gaussian noise as well as estimation of motion parameters for next frame which is used by modified LMedS algorithm. The estimated motion parameter for next frame is also used for calculation of estimated match points for features, which results in a small search area. By using this algorithm, the matching process is a two-frame based method, which doesn't suffer from problems such as non-uniform motions. The algorithm also uses the accumulated motion information of previous frames to calculate the motion model of current model and reduce the search area. Therefore the algorithm has taken the advantages of both two-frame based and long-sequence based methods.

4.1 Tracking Algorithm: Phase 1

The feature detection and matching algorithms of Sect. 2 are used to detect feature points of the target in the previous frame and find their corresponding points in current frame. To reduce the computational overhead of the matching process, we used a method, which uses the existing uniformity in the target motion to reduce the search area for matching. Initially uniform motion is assumed and the motion information accumulated up to current frame are used to estimate the locations of match points. Then small regions centered on these points are searched for the exact and subpixel matches. To detect non-uniformity in the motion, the percentage of matched features is calculated. If the percentage of matched features is low (typically lower than 60%), it means non-uniformity has occurred in the motion. In this case matching process is repeated using a larger search area and without using accumulated motion information. The accumulated motion information are also reset to the initial state in the case of non-uniform motion occurrence.

When the feature points and their matches are found, modified LMedS algorithm is utilized to calculate motion model parameters. Based on the reasons indicated in the detection algorithm, affine transformation is used for both phases of tracking algorithm. The input parameters for modified LMedS algorithm are the feature points and their matches as well as accumulated motion information, which is the estimated motion model for current frame and is calculated using Kalman Filter. In the modified LMedS algorithm instead of calculating one affine transformation, P affine transformations with the minimum median of squared differences, are calculated. Then the affine model which is the closest to estimated motion model, is selected. To do that, the target bonding-box of previous frame is transformed using the calculated models and estimated model. Subsequently the sum of squared differences between four corners of transformed bounding-boxes and estimated bounding-box are calculated. The closest model is the model, which has the minimum sum of squared differences. When there is no estimated motion model in the situations like mode switching and non-uniform motions, the algorithm uses LMedS algorithm to estimate affine parameters.

The state vector for the Kalman Filter in this phase is expressed as:

$$\begin{cases} X(k) = [\mathbf{a}(k), x_1(k), y_1(k), x_2(k), y_2(k)]^T \\ \mathbf{a}(k) = [a_1(k), a_2(k), a_3(k), a_4(k), a_5(k), a_6(k)] \end{cases} \tag{11}$$

where $a_1 - a_6$ are the affine parameters for target motion and (x_1, y_1), (x_2, y_2), represent the target bounding-box. Since rectangular bounding-box is considered, two points are sufficient to represent it. The state equations for the Kalman Filter in this phase are given as follows:

$$\begin{cases} \mathbf{a}(k+1) = \mathbf{a}(k) + \boldsymbol{\varepsilon}_1(k) \\ x_1(k+1) = a_1(k)x_1(k) + a_2(k)y_1(k) + a_5(k) + \varepsilon_2(k) \\ y_1(k+1) = a_3(k)x_1(k) + a_4(k)y_1(k) + a_6(k) + \varepsilon_3(k) \\ x_2(k+1) = a_1(k)x_2(k) + a_2(k)y_2(k) + a_5(k) + \varepsilon_4(k) \\ y_2(k+1) = a_3(k)x_2(k) + a_4(k)y_2(k) + a_6(k) + \varepsilon_5(k) \end{cases} \tag{12}$$

where $\boldsymbol{\varepsilon}(k)$ are called the process noise, which are zero mean and normally distributed Gaussian noise, and k is the time index. The measurement vector for Kalman Filter is the same as state vector. Therefore output equation can be expressed by the following equation:

$$\hat{\mathbf{X}}(k) = \mathbf{X}(k) + \boldsymbol{\zeta}(k) \tag{13}$$

where $\hat{\mathbf{X}}(k)$ are the measured outputs and $\boldsymbol{\zeta}(k)$ are measurement noise. The measured values for affine parameters are the output of modified LMedS algorithms. To obtain the measured values for target bounding-box i.e. $(\hat{x}_1, \hat{y}_1)$, $(\hat{x}_2, \hat{y}_2)$, the following algorithm is used.

1. Use affine parameters and calculate the position of the target bounding-box in the current frame by transforming the position of the target bounding-box in the previous frame.
2. Find and omit feature points which are inconsistent with the measured affine parameters. This helps removal of non-target feature points. The robust standard deviation estimate [21] is given by:

$$\hat{\sigma} = 1.4826[1 + 5/(N - p)]\sqrt{M_J} \tag{14}$$

where M_J is the minimal median, $p = 3$ for our algorithm and N is the total number of features used for calculation of the affine model. If the residual value of a feature point satisfies the equation $r_i^2 > 2.5\hat{\sigma}^2$, it is considered as inconsistent feature. Here r_i denotes the residual value.

3. Find the bounding-box, which covers all consistent match points. Since the corner detection algorithms don't consider boundary pixels, expand the bounding-box to compensate for corner detection and other pre-processing effects.
4. If $[(x_{t1}, y_{t1}), (x_{t2}, y_{t2})]$ and $[(x_{r1}, y_{r1}), (x_{r2}, y_{r2})]$ are the positions of the bounding-box obtained in step 1 and step 3 respectively, the measured position is calculated using the following equations:

$$\begin{cases} \hat{x}_i = x_{ti} + \alpha(x_{ri} - x_{ti}) \\ \hat{y}_i = y_{ti} + \alpha(y_{ri} - y_{ti}) \end{cases} \tag{15}$$

where α is a constant number smaller than 1 and $i = 1, 2$.

4.2 Tracking Algorithm: Phase 2

The second phase of the tracking algorithm is similar to the first phase. In this phase the affine parameters are calculated in the same manner as phase 1. However the equations for the Kalman Filter are different. In this phase the filter state is simply the affine parameters and their derivatives, i.e. $[\mathbf{a}, \dot{\mathbf{a}}]$, and constant velocity state dynamic is used as follows:

$$\begin{pmatrix} \mathbf{a}(k+1) \\ \dot{\mathbf{a}}(k) \end{pmatrix} = \begin{pmatrix} \mathbf{a}(k) + \dot{\mathbf{a}}(k) \\ \dot{\mathbf{a}}(k) \end{pmatrix} + \boldsymbol{\varepsilon}(k) \tag{16}$$

The filter measurements in this phase are only the affine parameters, which are calculated using modified LMedS algorithm and assumed to be the noisy version of true parameters, i.e.

$$\hat{\mathbf{a}}(k) = \mathbf{a}(k) + \boldsymbol{\zeta}(k) \tag{17}$$

where $\boldsymbol{\zeta}$(k) are normally distributed Gaussian noise and $\hat{\mathbf{a}}(k)$ are the measured affine parameters and are calculated using modified LMedS algorithm.

5. Experimental Results

The algorithm has been implemented on a Pentium III 500 Mhz under Windows 98 operating system using a Visual C++ program. The algorithm has been tested with both simulated and actual sequences of images for objects and vehicles in different landscapes. The system can detect and track targets in real-time. We tested the proposed algorithms with wide variety of image sequences. Figure 5 shows some results for detection of various objects in arbitrary backgrounds. As it is shown the algorithm has successfully detected the targets. Our results showed that the threshold value of 0.15 for V and H (see Sect. 3.2) suppresses most of non-vehicle targets. Figure 6 shows the Car image sequence used for testing the proposed tracking algorithms as well as feature detection and matching algorithms. To evaluate the tracking algorithms, normalized tracking error is calculated as a function of time. The normalized tracking error is given by the following equation:

$$e = \left((x_{1e}-x_{1r})^2+(y_{1e}-y_{1r})^2+(x_{2e}-x_{2r})^2+(y_{2e}-y_{2r})^2\right) \Big/ \left((x_{1r} - x_{2r})^2 + (y_{1r} - y_{2r})^2\right) \tag{18}$$

where $((x_{1e}, y_{1e}),\ (x_{2e}, y_{2e}))$ are the location of calculated target bounding-box by tracking algorithm and $((x_{1r}, y_{1r}),\ (x_{2r}, y_{2r}))$ are the real location of target bounding-box. To compare the efficiency of the different matching algorithms, matching percentage is calculated where this percentage is given by:

Fig. 5 Various targets detected using the detection algorithm.

Fig. 6 Different frames of Car image sequence. Rectangular regions with inverted color show the region of the target tracked by the tracking algorithm phase 2.

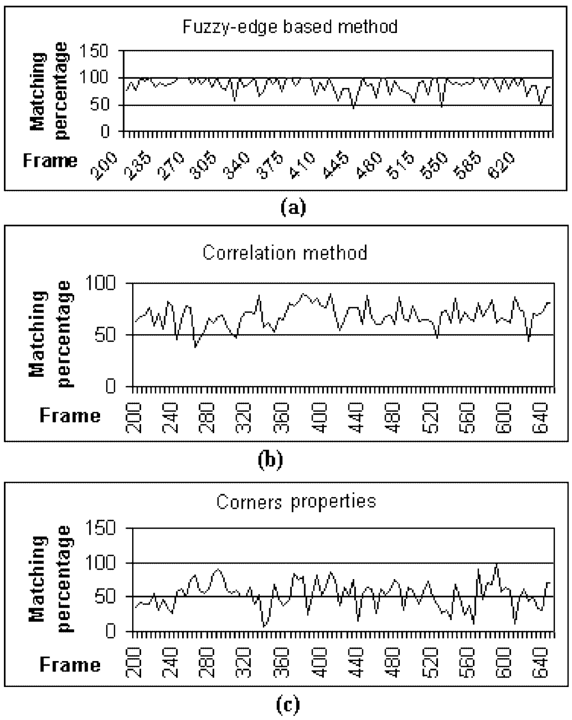


Fig. 7 Matching percentage for different frames of Car image sequence of Fig. 6, (a) fuzzy-edge based method (b) correlation method (c) matching using corners properties.

$$MP = \frac{CM - IM}{MPCM} \tag{19}$$

where MP is the matching percentage, CM is the number of correct matches, IM is the number of incorrect matches and $MPCM$ is the maximum possible number of correct matches.

The proposed algorithms were also tested with noisy images and images with illumination change. To generate images with illumination change, the following equation is used:

$$I_{ill}(x,y) = (ax + by + c)I(x,y) + (dx + ey + f) \tag{20}$$

where $I(x,y)$ are input images, a,b,c,d,e,f are random numbers, which are selected differently for each frame and $I_{ill}(x,y)$ are the images with illumination changes.

Figure 7 depicts the matching percentage for different frames of the image sequence of Fig. 6. The results of three matching algorithms are shown in this figure, which are: 1- the proposed edge based feature detection and matching algorithms 2- SUSAN corner detector and correlation based feature matching 3- matching using SUSAN corners properties [19]. As it is shown in this figure the proposed edge feature detection and matching algorithms have the best results. To show the efficiency of the proposed feature detection and matching algorithms in the presence of image noise and illumination change, the above experiment was repeated with additive Gaussian noise of 10 dB and illumination change simulation. Table 1 shows the average matching perecentages of different matching algorithms in the case of noisy images or images with illumination change. As it is shown in Table 1, in all cases the proposed feature detection and matching algorithms give appropriate results. Also the comparison of fuzzy-edge based method with edge correlation using Car image sequence of Fig. 6 showed that the fuzzy-edge based method enhances the average matching percentage up to 15.

Table 1 Average matching percentage for Car image sequence of Fig. 6 using different matching algorithms, Image type: 1- without noise and illumination change, 2- with Gaussian noise of 10 dB 3- with illumination change.

Image Type	Matching Algorithms		
	Corners properties	Correlation	Fuzzy-edge based
1	53.48	68.72	86.21
2	−15.38	45.13	70.02
3	32.37	64.97	71.49

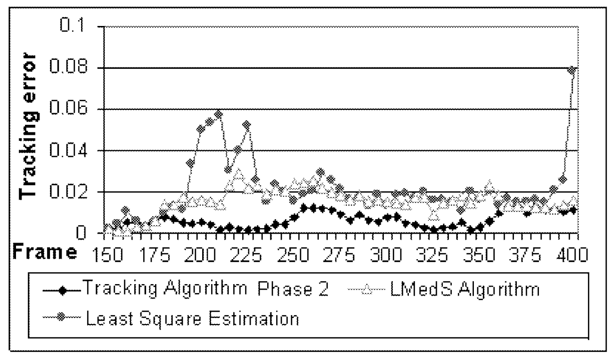


Fig. 8 Tracking error for different frames of Car image sequence of Fig. 6.

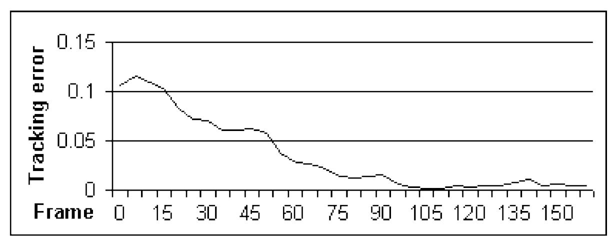


Fig. 9 The performance of tracking algorithm phase 1. We intentionally selected a part of the vehicle bounding-box tracked in Fig. 8 as the target. Tracking algorithm phase 1, has gradually recovered the entire contour of the vehicle after a few frames.

Figure 8 shows the results of tracking algorithms for image sequence of Fig. 6. For comparison purpose, the results of tracking are depicted for three different tracking algorithms including tracking algorithm phase 2, LMedS estimator and least mean square estimator. As it is shown in Fig. 4, the efficiency of tracking algorithm phase 2 is better than other algorithms. To show the efficiency of tracking algorithm phase 1, we intentionally selected part of the vehicle tracked in Fig. 8, as the target. As it is depicted in Fig. 9, the tracking algorithm phase 1, has gradually recovered the complete contour of the target.

Figure 10 shows another image sequence, which is used for comparison of different matching and tracking algorithms. This image sequence has lower motion speed compared to image sequence of Fig. 6. The average matching percentages of different matching algorithms are shown in Table 2. Figure 11 shows the tracking errors for different frames of Fig. 10 with additive Gaussian noise of 7 dB.

If the target has aspect change so that some part of the target appears or disappears, tracking algorithm phase 1 can deal with this situation because this algorithm tries to detect full contour of the target using its motion model. However tracking algorithm phase 2 assumes that the target has no aspect change. Figure 12 shows the results of tracking algorithm phase 2 for a target with aspect change. As it is shown in this figure, the aspect change of the target is not supported. To handle aspect change in the tracking algorithm phase 2, we propose that the contour of the target is updated using any contour detection algorithm [6], [24] after the determination of the target bounding-box. The results of the tracking algorithm phase 2 with this modification are shown in Fig. 13. In this experiment we used the edge contours used for the detection of edge feature for detection of the target contour.

It is important to note that, this modification enables the tracking algorithm phase 2 to handle aspect change. However in the occluded targets it may reduce the efficiency of the original algorithm because the contour detection algorithm may not detect the contour of the target precisely.

The proposed algorithms were also tested with other image sequences and different targets. Results showed the accuracy of the methods in detecting and tracking of moving objects. Comparison of the results generated by the proposed method with those of other

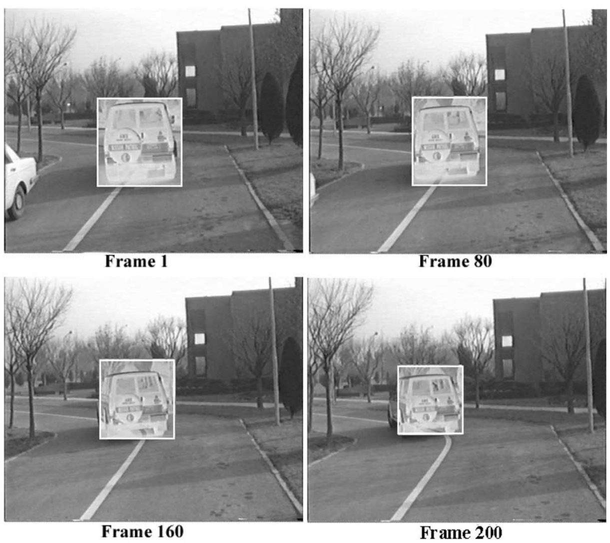


Fig. 10 Different frames of Patrol image sequence. Rectangular regions with inverted color show the region of the target tracked by the tracking algorithm phase 2 with additive Gaussian noise of 7 dB.

Table 2 Average matching percentage for Patrol image sequence of Fig. 10 using different matching algorithms, Image type: 1- without noise and illumination change, 2- with Gaussian noise of 10 dB, 3- with illumination change.

Image Type	Matching Algorithms		
	Corners properties	Correlation	Fuzzy-edge based
1	63.46	92.05	95.51
2	30.31	82.00	93.79
3	34.36	89.25	81.09

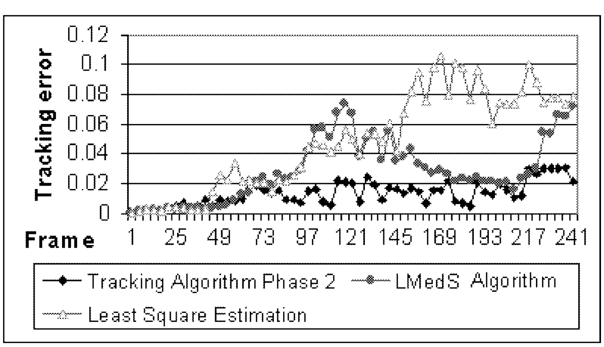


Fig. 11 Tracking error for different frames of Patrol image sequence of Fig. 10 with additive Gaussian noise of 7 dB.

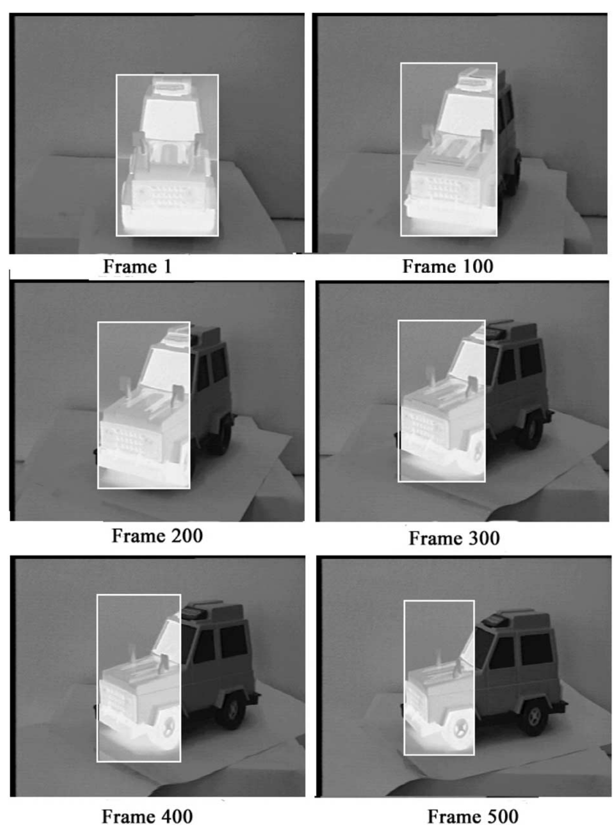


Fig. 12 Tracking result for different frames of Toy image sequence using tracking algorithm phase 2.

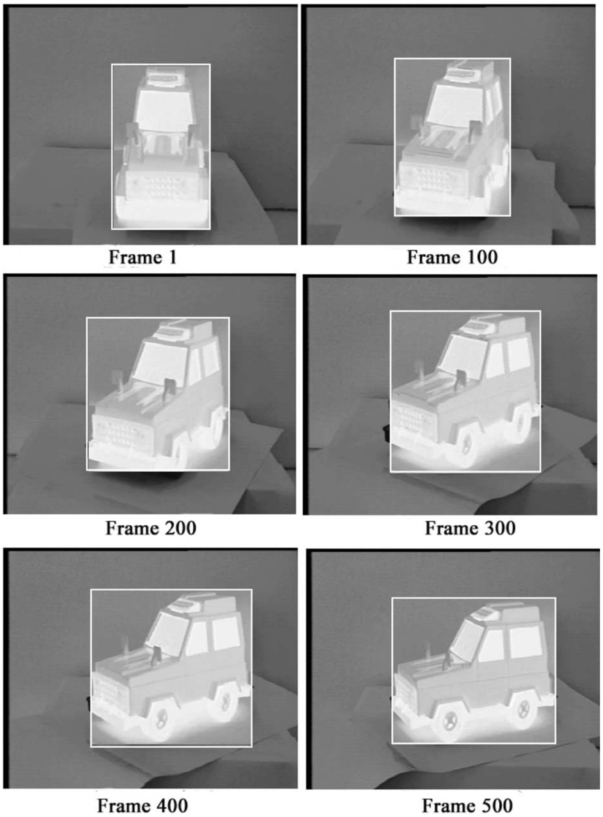


Fig. 13 Tracking result for different frames of Toy image sequence using tracking algorithm phase 2 and contour update.

methods showed that more reliable results could be obtained with the aid of the proposed methods.

6. Conclusions

In this paper we proposed a new method for detection and tracking of the moving objects using mobile camera. We used two different methods for detection and tracking. For detection, we used affine transformation and LMedS method for estimation of the apparent background motion. When the apparent motion of the background is cancelled, the difference of two consecutive frames is used for detection of the moving target. For robust detection of moving target, we also checked the target for some features. Our tracking algorithm contains two phases. We used modified LMedS algorithm and Kalman Filtering for fast and robust tracking purpose for both phase of tracking algorithm. The utilized methods take the advantages of both the two-frame based and long-sequence based methods. We used edge points and their accumulated curvature information for feature detection. The detected edge features are then matched with their corresponding points using a new method called fuzzy-edge based feature matching.

We tested our algorithms with wide variety of image sequences. The proposed methods successfully detect and track moving vehicles and objects in arbitrary scenes obtained from a mobile video camera. The tracking system is capable of target shape recovery and therefore it can successfully track targets with varying distance from camera or while the camera is zooming. Experimental results have shown that the algorithm is reliable and can successfully detect and track targets in most cases and in environments with noise and illumination change.

References

[1] A.J. Lipton, F. Fujiyoshi, and R.S. Patil, "Moving target classification and tracking from real-time video," Proc. IEEE Workshop Application of Computer Vision, pp.8–14, Oct. 1998.

[2] S. Araki, T. Matsuoaka, N. Yokoya, and H. Takemura, "Real-time tracking of multiple moving object contours in a moving camera image sequence," IEICE Trans. Inf. & Syst., vol.E83-D, no.7, pp.1583–1591, July 2000.

[3] D. Murray and A. Basu, "Motion tracking with an active camera," IEEE Trans. Pattern Anal. Mach. Intell., vol.16, no.5, pp.449–459, May 1994.

[4] J.L. Barron, D.J. Fleet, S.S. Beauchemin, and T.A. Burkitt, "Performance of optical flow techniques," Int. J. Comput. Vis., vol.12, no.1, pp.43–77, 1992.

[5] B.K.P. Horn and B.G. Schunck, "Determining optical flow," Artif. Intell., vol.17, pp.185–203, Aug. 1981.

[6] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," Int. J. Comput. Vis., vol.1, no.4, pp.321–331, Jan. 1987.

[7] L.D. Cohen, "On active contour models and balloons," Computer Vision, Graphics and Image Processing: Image Understanding, vol.53, no.2, pp.211–218, 1991.

[8] A. Fusiello, E. Trucco, T. Tommasini, and V. Roberto, "Improving feature tracking with robust statistics," Springer-Verlag Pattern Analysis & Applications, vol.2, no.4, pp.312–320, 1999.

[9] S.D. Blostein and T.S. Huang, "Detecting small moving objects in image sequences using sequential hypothesis testing," IEEE Trans. Signal Process., vol.39, pp.1611–1629, July 1991.

[10] L.K. Sethi and R. Jain, "Finding trajectories of feature points in a monocular image sequences," IEEE Trans. Pattern Anal. Mach. Intell., vol.9, no.1, pp.56–73, Jan. 1987.

[11] J. Weng, N. Ahuja, and T.S. Huang, "Matching two percepective views," IEEE Trans. Pattern Anal. Mach. Intell., vol.14, no.8, pp.806–825, Aug. 1992.

[12] Y.S. Yao and R. Chellappa, "Tracking a dynamic set of feature points," IEEE Trans. Image Process., vol.4, no.10, pp.1382–1395, Oct. 1995.

[13] J. Canny, "A computational approach to edge detection," IEEE Trans. Pattern. Anal. Mach. Intell., vol.8, no.6, pp.679–698, 1986.

[14] F. van der Heijden, "Edge and line feature extraction based on covariance models," IEEE Trans. Pattern Anal. Mach. Intell., vol.17, no.1, pp.16–33, Jan. 1995.

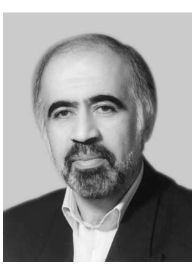
[15] H.P. Moravec, "Towards automatic visual obstacle avoidance," Proc. 5th International Joint Conference on Artificial Intelligence, p.584, Cambridge, MA, USA, 1977.

[16] C.G. Harris and M. Stephens, "Combined corner and edge detector," Proc. 4th Alvey Vision Conference, pp.147–151, Manchester, England, 1988.

[17] S.M. Smith and J.M. Brady, "SUSAN - A new approach to

low level image processing," Int. J. Comput. Vis., vol.23, no.1, pp.45–78, May 1997.
[18] F. Mokhtarian and R. Suomela, "Robust image corner detection through curvature scale space," IEEE Trans. Pattern Anal. Mach. Intell., vol.20, no.12, pp.1376–1381, Dec. 1998.
[19] S.M. Smith and J.M. Brady, "ASSET-2: Real-time motion segmentation and shape tracking," IEEE Trans. Pattern Anal. Mach. Intell., vol.17, no.8, pp.814–820, Aug. 1995.
[20] L.X. Wang, A Course in Fuzzy Systems and Control, Prentice-Hall International Inc., 1997.
[21] P.J. Rousseeuw and A.M. Leroy, Robust Regression and Outlier Detection, John Wiley & Sons, New York, 1987.
[22] P.H.S Torr and D. Murray, "Outlier detection and motion segmentation," Sensor Fusion VI, SPIE, vol.2059, pp.432–443, 1993.
[23] M.A. Fischler and R.C. Bolles, "Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography," Commun. ACM, vol.24, pp.381–395, 1981.
[24] L.H. Staib and J.S. Duncan, "Boundary finding with parametrically deformable models," IEEE Trans. Pattern Anal. Mach. Intell., vol.14, no.11, pp.1061–1075, Nov. 1992.

Alireza Behrad was born in Iran in 1973. He received the B.S. degree in electronic engineering from Tabriz University, Tabriz, Iran, in 1995. In 1998, he received M.S. degree in digital electronic from Sharif University of Technology, Tehran, Iran. Currently he is the Ph.D. candidate of Amirkabir University of Technology, Tehran, Iran. His research fields are Image processing, Target Tracking using video image sequence and ECG signal processing.



Seyed Ahmad Motamedi received the B.S. degree in Electronic Engineering from Amirkabir University of Technology, Tehran, Iran, in 1979. He received the M.S. degree in Computer Hardware in 1981 and Ph.D. degree in Informatic Systems (Computer Hardware) in 1984, both from University of Pierre & Marie Curie (Paris VI), France. Currently he is an associate Prof. in Electrical Engineering Department, Amirkabir University of Technology. He was the president of Iranian Research Organization for science and Technology (IROST) from 1986 to 2001. His research interests include, Image processing, Parallel Processing, Microprocessor Systems, Automation, Biomedical Engineering. He has more than 60 publications in international conference proceedings and scientific journals. He is also the author of 3 books.