Available online at www.sciencedirect.com
ScienceDirect
Image and Vision Computing xxx (2006) xxx–xxx
www.elsevier.com/locate/imavis

Robust tracking with motion estimation and local Kernel-based color modeling ☆

R. Venkatesh Babu [a,*], Patrick Pérez [b], Patrick Bouthemy [b]

[a] *Department of Electrical Engineering, Indian Institute of Science, Bangalore, India*
[b] *IRISA/INRIA-Rennes, France*

Received 9 February 2006; received in revised form 15 June 2006; accepted 12 July 2006

Abstract

Visual tracking has been a challenging problem in computer vision over the decades. The applications of visual tracking are far-reaching, ranging from surveillance and monitoring to smart rooms. Mean-shift tracker, which gained attention recently, is known for tracking objects in a cluttered environment. In this work, we propose a new method to track objects by combining two well-known trackers, sum-of-squared differences (SSD) and color-based mean-shift (MS) tracker. In the proposed combination, the two trackers complement each other by overcoming their respective disadvantages. The rapid model change in SSD tracker is overcome by the MS tracker module, while the inability of MS tracker to handle large displacements is circumvented by the SSD module. The performance of the combined tracker is illustrated to be better than those of the individual trackers, for tracking fast-moving objects. Since the MS tracker relies on global object parameters such as color, the performance of the tracker degrades when the object undergoes partial occlusion. To avoid adverse effects of the global model, we use MS tracker to track local object properties instead of the global ones. Further, likelihood ratio weighting is used for the SSD tracker to avoid drift during partial occlusion and to update the MS tracking modules. The proposed tracker outperforms the traditional MS tracker as illustrated.

Keywords: Visual tracking; Mean-shift; Object tracking; Kernel tracking

1. Introduction

The objective of object tracking is to faithfully locate the targets in successive video frames. The major challenges encountered in visual tracking are cluttered background, noise, change in illumination, occlusion and scale/appearance change of the objects. Considerable work has already been done in visual tracking to address the aforementioned challenges. Most of the tracking algorithms can be broadly classified into the following four categories.

(1) *Gradient-based methods* locate target objects in the subsequent frame by minimizing a cost function [1,2].
(2) *Feature-based approaches* use features extracted from image attributes such as intensity, color, edges and contours for tracking target objects [3–5].
(3) *Knowledge-based tracking algorithms* use *a priori* knowledge of target objects such as shape, object skeleton, skin color models and silhouette [6–9].
(4) *Learning-based approaches* use pattern recognition algorithms to learn the target objects in order to search them in an image sequence [10–12].

Visual tracking in a cluttered environment remains one of the challenging problems in computer vision for the past few decades. Various applications like surveillance and monitoring, video indexing and retrieval require the ability to faithfully track objects in a complex scene involving appearance and scale change. Though there exist many

☆ This work was supported by an ERCIM post-doctoral fellowship at IRISA/INRIA, Rennes, France.
* Corresponding author. Tel.: +91 80 23722879.
E-mail addresses: venkatesh.babu@gmail.com (R.V. Babu), perez@irisa.fr (P. Pérez), Patrick.Bouthemy@irisa.fr (P. Bouthemy).

0262-8856/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2006.07.016

techniques for tracking objects, color-based tracking with kernel density estimation, introduced in [13,8], has recently gained more attention among research community due to its low computational complexity and robustness to appearance change. The reported work in [13] is due to the use of a deterministic gradient ascent (the "mean shift" iteration) starting at the location, corresponding to the object location in previous frame. A similar work in [8] relies on the use of a global appearance model, e.g., in terms of colors, as opposed to very precise appearance models such as pixel-wise intensity templates [14,15].

The mean-shift algorithm was originally proposed by Fukunaga and Hostetler [16] for clustering data. It was introduced to image processing community by Cheng [17] a decade ago. This theory became popular among vision community after its successful application to image segmentation and tracking by Comaniciu and Meer [18,5]. Later, many variants of the mean-shift algorithm were proposed for various applications [19–24].

Though mean-shift tracker performs well on sequences with relatively small object displacement, its performance is not guaranteed when the objects move fast as well as when they undergo partial occlusion. Here, we attempt to improve the performance of mean-shift tracker when the object undergoes large displacements (when the object regions do not overlap between the consecutive frames) and in the event of partial/full occlusion. The problem of large displacements is tackled by cascading an SSD tracker with the mean-shift tracker. An SSD tracker based on frame-to-frame appearance matching, is useful in finding the object location in successive frames. However, the problem with SSD tracker is its short-term memory which can cause drifting problems or even complete loss in worse cases. On the other hand, MS trackers which rely on persistent global object properties such as color, can be much more robust to detailed appearance changes due to shape and pose changes. However, MS tracker has problems with large displacements. It thus seems interesting to combine the advantages of the two aforementioned trackers.

In order to improve the performance of MS tracker, in the event of the object undergoing partial occlusion, we propose to rely on a number of elementary MS modules (tracking points) embedded within the object, rather than on a single global MS tracker representing the whole object. We also address the issue of large scale changes due to camera operations.

For each of the above-mentioned challenges, solutions proposed so far have been within the realm of pure MS trackers: incorporation of a dynamic model (e.g., using Kalman filter in [13,25] or particle filter in [26,27]) to cope with large displacements, occlusions and, to some extent, with scale changes; simple linear histogram updates with fixed forgetting factor [27] for on-line adaptation of reference model; rather complex procedures [28,29] for addressing the generic problem of scale changes (independent of their origin).

The novelty of the proposed approach lies in a one-step approach which exploits the fact that the reference color model and instantaneous motion estimation based on pixel-wise intensity conservation, complement one another. The latter is provided by greedy minimization of the intensity sum-of-squared differences (SSD), which is classic in point tracking and motion field estimation, by block matching. Scale changes of the object that are due to camera zoom effect or ego-motion are estimated by approximating the dominant apparent image motion by an affine model. By using local object color models (tracking points embedded on the object) instead of a global one, the performance of the tracker is greatly improved when the object undergoes partial occlusion.

The paper is organized as follows. Section 2 explains the proposed SSD/MS combined tracker to track fast moving objects. The problem of occlusion handling is discussed in Section 3. The results illustrating the performance of proposed tracker are given in Section 4. Concluding remarks are given in Section 5.

2. Proposed combined tracker

In this work, tracking is done in Kalman filter framework. The object to be tracked is specified by the location of its center and scale (for a fixed aspect ratio) in the image plane. The objective of the tracking algorithm is to find the object location in successive frames. In this work, we cascade SSD tracker with MS tracker to obtain better tracking performance. The measurements obtained by the combined tracker module are used for estimating the states of the Kalman filter. The overview of the proposed system is illustrated in Fig. 1.

The state-space representation of the tracker used in Kalman filter framework is given below:

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ x_t \\ y_t \\ s_{t+1} \end{bmatrix} = \begin{bmatrix} 2 & 0 & -1 & 0 & 0 \\ 0 & 2 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x_t \\ y_t \\ x_{t-1} \\ y_{t-1} \\ s_t \end{bmatrix} + \mathbf{w}_t, \quad (1)$$

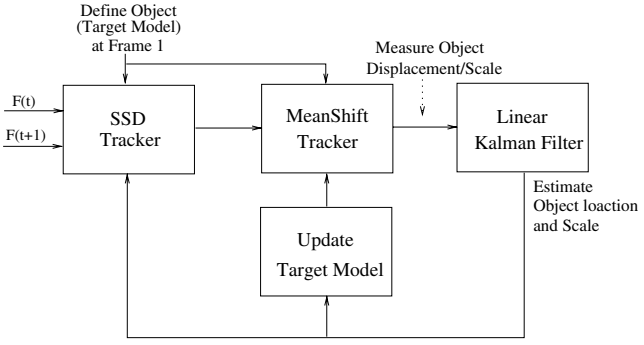

Fig. 1. Overview of the proposed tracking system.

where, $\mathbf{x}_t = (x_t, y_t)$ indicates the location of the object center at time t, s_t represents the scale at time t and $\mathbf{w}_t$ is white Gaussian noise with diagonal variance Q. The measurement equation relates the states and measurements at time t as follows:

$$\begin{bmatrix} u_t \\ v_t \\ \xi_t \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x_t \\ y_t \\ x_{t-1} \\ y_{t-1} \\ s_t \end{bmatrix} + \mathbf{z}_t, \tag{2}$$

where, $\mathbf{u}_t = (u_t, v_t)$ is the measured velocity (displacement) of the object, ξ_t is the measured scale at time t, and $\mathbf{z}_t$ is white Gaussian noise with diagonal variance R. The displacement measurement $\mathbf{u}_t$ is obtained through the SSD-MS tracker module, whereas scale measurement is provided by global parametric motion estimation.

2.1. SSD-MS motion measurement

SSD tracker localizes the object in a given search window of the successive frame based on minimum distance criterion between the target and candidate object images. SSD tracker works well even for large displacements as long as the object appearance changes only slightly between two adjacent frames. However, in reality, the appearance of the object often changes considerably with time. In a typical SSD tracker, the winning candidate becomes the new target for the next time instance. This process might make the SSD tracker forget the original model with time, although it performs well between any two consecutive frames for a given target.

Given the state estimate $(\hat{\mathbf{x}}_{t-1}, \hat{s}_{t-1})$ at previous instant, the SSD-based displacement estimate is

$$\mathbf{u}_t^{\text{ssd}} = \arg\min_{\mathbf{u}\in W} \sum_{\mathbf{d}\in D} [F_t(\mathbf{u} + \hat{\mathbf{x}}_{t-1} + \hat{s}_{t|t-1}\mathbf{d}) - F_{t-1}(\hat{\mathbf{x}}_{t-1} + \hat{s}_{t-1}\mathbf{d})]^2, \tag{3}$$

where, F_{t-1} and F_t are the two consecutive intensity images, $\hat{s}_{t|t-1} = \hat{s}_{t-1}$ is the scale prediction, W is the search window, and D is the normalized sub-image support (rectangle of the size of original object with the origin placed at its center).

This first displacement estimate is used for initializing the MS tracker. The target color model $\mathbf{q} = (q_i)_{i=1\ldots m}$, with $\sum_{i=1}^m q_i = 1$, is composed of m bins in some appropriate color space (e.g., RGB or Hue-Saturation). The bins are constructed at the start of the tracking. The candidate histogram $\mathbf{p}(\mathbf{x}, s) = (p_i(\mathbf{x},s))_{i=1\ldots m}$, at location $\mathbf{x}$ and scale s in the current frame is given by:

$$p_i(\mathbf{x}, s) = \frac{\sum_{\mathbf{d}\in s\cdot D} k(s^{-2}|\mathbf{d}|^2)\delta[b(\mathbf{x}+\mathbf{d}) - i]}{\sum_{\mathbf{d}\in s\cdot D} k(s^{-2}|\mathbf{d}|^2)}, \tag{4}$$

where, $k(x)$ is a convex and monotonically decreasing Epanechnikov kernel profile, almost everywhere differentiable and with support D, which assigns smaller weights to pixels far away from the center, δ is the Kronecker delta function, and function $b(\mathbf{x}) \in \{1 \ldots m\}$ is the color bin number at pixel $\mathbf{x}$ in the current frame. One seeks the location whose associated candidate histogram is as similar as possible to the target one. When similarity is measured by Bhattacharyya coefficient, $\rho(\mathbf{p}, \mathbf{q}) = \sum_i \sqrt{p_i q_i}$, convergence towards the nearest local minima is obtained by the iterative mean-shift procedure [13]. In our case, this gradient ascent at time t is initialized at $\mathbf{y}_0 = \hat{\mathbf{x}}_{t-1} + \mathbf{u}_t^{\text{ssd}}$ and proceeds as follows:

(1) Given current location $\mathbf{y}_0$ and scale s, compute candidate histogram $\mathbf{p}(\mathbf{y}_0,s)$ and Bhattacharyya coefficient $\rho[\mathbf{p}(\mathbf{y}_0,s),\mathbf{q}]$.
(2) Compute candidate position.

$$\mathbf{y}_1 = \frac{\sum_{\mathbf{d}\in s\cdot D} w(\mathbf{y}_0 + \mathbf{d})k'(s^{-2}|\mathbf{d}|^2)(\mathbf{y}_0 + \mathbf{d})}{\sum_{\mathbf{d}\in s\cdot D} w(\mathbf{y}_0 + \mathbf{d})k'(s^{-2}|\mathbf{d}|^2)}$$

with weights at location $\mathbf{x}$

$$w(\mathbf{x}) = \sum_{i=1}^m \sqrt{\frac{q_i}{p_i(\mathbf{y}_0, s)}}\delta[b(\mathbf{x}) - i],$$

where, k' is derivative of kernel profile k.

While $\rho[\mathbf{p}(\mathbf{y}_1,s),\mathbf{q}] < \rho[\mathbf{p}(\mathbf{y}_0,s),\mathbf{q}]$ do $\mathbf{y}_1 \leftarrow \frac{1}{2}(\mathbf{y}_1 + \mathbf{y}_0)$.
If $\|\mathbf{y}_1 - \mathbf{y}_0\| < \varepsilon$ then stop, otherwise set $\mathbf{y}_0 \leftarrow \mathbf{y}_1$ and repeat Step 2.

The final estimate provides the displacement measurement $\mathbf{u}_t = \mathbf{y}_1 - \hat{\mathbf{x}}_{t-1}$. Finally, the two entries associated with this measurement in the covariance matrix R_t of the observation model (2) are chosen as

$$\sigma_u^2 = \sigma_v^2 = \alpha e^{-\beta\kappa(\mathbf{y}_1)}, \tag{5}$$

where, $\kappa(\mathbf{y}_1)$ is the curvature of SSD function around $\mathbf{y}_1$ and α and β are two parameters set to *100* and 50 respectively in the experiments.

2.2. Scaling measurement

Scaling is a very important parameter in visual tracking. Often scale changes of the objects are due to camera zoom operation or camera ego-motion. The scale change in our work is measured (to be plugged in Kalman Filter) through the affine motion parameters of the global (dominant) image motion between the current and subsequent frames. Quick and robust estimation of these parameters is obtained using [30]. If the 2×2 matrix A_t stands for the linear part of the affine motion model thus estimated at time t, the measured zoom factor is

$$\xi_t = 1 + 0.5\ \mathrm{trace}(A_t). \quad (6)$$

The entry associated with this measurement in the covariance matrix R_t of the observation model (2) is set to a small constant (1 in the experiments).

2.3. Target model update

Updating the target model is one of the crucial issues in tracking. The performance of the mean-shift algorithm decreases considerably when the global appearance of the object changes with time. In this case, the color histogram obtained using the target definition from the first frame correlates less with the current view of the tracked object. In order to maintain the effectiveness of the mean-shift tracker in this scenario, it is essential to update the target model while tracking. This model update helps the tracker perform well in cluttered background conditions and in the event of appearance changes. In our system, Bhattacharyya distance, which measures the distance between target model and the one at the current location estimate provided by the Kalman filter, is used to update the reference. This model update is a trade-off between adaptation to rapid changes and robustness to changes due to occlusion. In our system, candidate models close to the target contribute more than farther ones. The update procedure used is defined as:

$$\mathbf{q}_{t+1} = \mathbf{q}_t + \mathrm{e}^{-\alpha[1-\rho(\mathbf{q}_t, \mathbf{p}(\hat{\mathbf{x}}_t, \hat{s}_t))]}\mathbf{p}(\hat{\mathbf{x}}_t, \hat{s}_t), \quad (7)$$

where, α is a real positive scalar, which determines the model update rate. Typical value of α used in our experiments is set to 10.

The performance of the proposed combined-tracker is discussed in Section 4.1.

3. Occlusion handling

Although the proposed combined tracker works better than individual SSD or MS tracker, its performance degrades when the object undergoes partial occlusion for a longer time duration. This is clearly illustrated in Fig. 3.

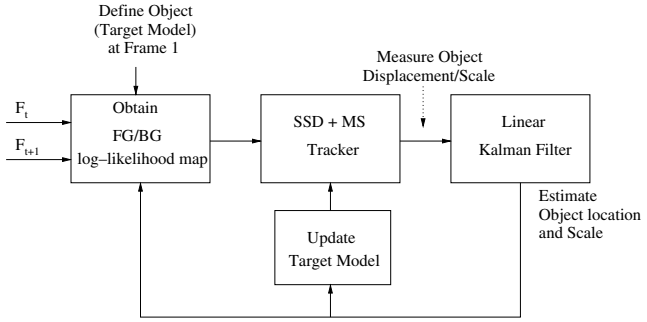

Fig. 2. Overview of the proposed tracking system with occlusion handling.

Here, the MS tracker (green window) shifts up to due to partial occlusion, though there is no upward displacement of the object. This shift is due to the property of the MS tracker which maximizes the similarity (Bhattacharya coefficient) between the target model under no occlusion and the candidate model under partial occlusion. Since this adverse effect of MS tracker is due to the single tracking window that represents the object, we embed a number of elementary tracking windows (tracking points) on the object within the global MS tracker window. Now the object is tracked using the information only from the reliable MS tracking points that are not undergoing occlusion. The overview of this occlusion handling process is illustrated in Fig. 2. The following sections explain each of the modules in detail.

3.1. Object-background separation and initialization of tracking points

Tracking an object undergoing partial occlusion can be performed if we can separate the object region from the background at each time instant. The object-background separation is useful in weighting the pixels for SSD tracker and helps to locate reliable MS modules for updating. To achieve this, the R-G-B based joint pdf of the object region and that of a neighborhood surrounding the object is obtained. This process is illustrated in Fig. 4. The region within the red rectangle is used to obtain the object pdf

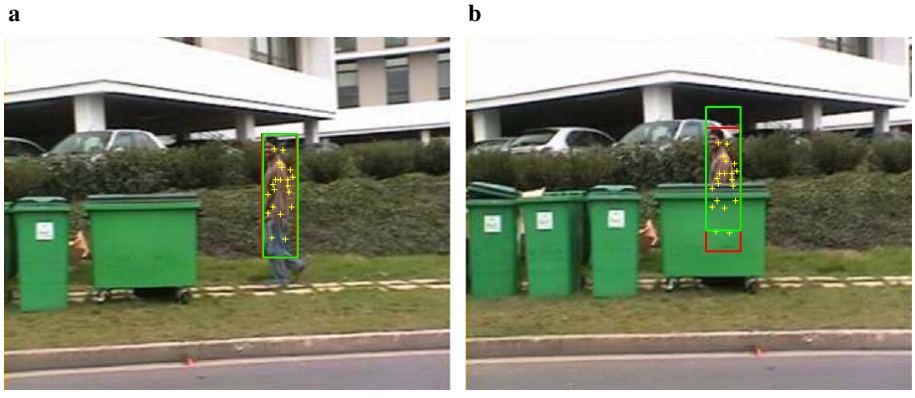

Fig. 3. The shift of tracking window when the object undergoes partial occlusion. In (b), the red window is the desired tracking window, the green window is shifted up due to partial occlusion.

Please cite this article as: R. Venkatesh Babu et al., Robust tracking with motion estimation and ..., Image and Vision Computing (2006), doi:10.1016/j.imavis.2006.07.016

Fig. 4. Track point initialization using T_0: (a) Initial frame with object boundary (b) likelihood map T_0 (c) Mask obtained after morphological operations (d) tracking points with the support region. Here, the number of tracking points is 20 with a support region of 100 pixels.

and the region between the green and red rectangles is used for obtaining the background pdf. The resulting log-likelihood ratio of foreground/background region is used to determine object pixels. The log-likelihood of a pixel considered, at time t, within the outer bounding rectangle (green rectangle in Fig. 4) is obtained as

$$L_t(i) = \log \frac{\max\{h_o(i), \epsilon\}}{\max\{h_b(i), \epsilon\}}, \tag{8}$$

where, $h_o(i)$ and $h_b(i)$ are the probabilities of ith pixel belonging to the object and background respectively; and ϵ is a small non-zero value to avoid numerical instability. The non-linear log-likelihood function maps the multimodal object/background distribution as positive values for colors associated with foreground, while negative values are marked for background. Only reliable object pixels are used as weighting factors for SSD tracker. The weighting factor T_t is obtained as:

$$T_t(i) = \begin{cases} L_t(i) & \text{if } L_t(i) > th_o, \\ 0 & \text{otherwise}, \end{cases} \tag{9}$$

where, th_o is the threshold to decide on the most reliable object pixels. In our experiments the value of th_o is set at 0.8. Once the object is localized, by user interaction or detection in the first frame, the tracking points are placed on the object in the first frame. Then the likelihood map of the object/background is obtained using (9). A binary mask corresponding to T_t is obtained by mapping all positive values of T_t to 1. This object mask is further subjected to morphological closing operation and used for embedding the tracking points (see Fig. 4). The tracking points are randomly spread, with due care, to ensure that their center lies within the object mask. In our experiments, the support region of all tracking points is a square of side $c \cdot \min(\textit{object length}, \textit{object width})$. The typical range of c used in our experiments is 0.3 to 0.5.

3.2. Tracker with occlusion handling

Given the state estimate $(\hat{\mathbf{x}}_{t-1}, \hat{s}_{t-1})$ at previous instant, the modified SSD-based displacement estimate now is:

$$\mathbf{u}_t^{\text{ssd}} = \arg\min_{\mathbf{u}\in W} \sum_{\mathbf{d}\in D} T_t.[F_t(\mathbf{u} + \hat{\mathbf{x}}_{t-1} + \hat{s}_{t|t-1}\mathbf{d}) - F_{t-1}(\hat{\mathbf{x}}_{t-1} + \hat{s}_{t-1}\mathbf{d})]^2 \tag{10}$$

where, T_t is the weighting function obtained from the foreground/background likelihood maps.

In our work, instead of using a single MS tracker for the entire object, we use multiple small regions of the object for tracking. The locations of these tracking points are randomly placed on the object area with the help of previously obtained object/background likelihood maps.

The first displacement estimate given by (10) is used for initializing these MS trackers. Let N be the total number of tracking points. The target color models

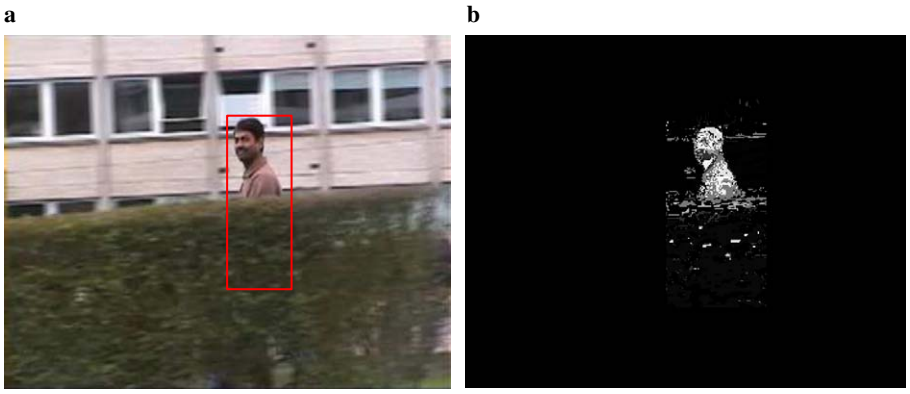

Fig. 5. (a) One frame showing object under partial occlusion and the (b) corresponding log-likelihood map.

$\mathbf{q}^n = (q_i^n)_{i=1\cdots m}$, with $\sum_{i=1}^{m} q_i^n = 1$, are composed of m bins in some appropriate color space (e.g., RGB or Hue-Saturation, in our experiments RGB color space with 10 bins along each dimension is used), where superscript $n \in N$ indicates the nth model corresponding to the nth tracking point. The target model bins are constructed at the start of the tracking. The candidate histogram $\mathbf{p}^n$, at location $\mathbf{x}^n$ and scale s in the current frame is given by:

$$p_i^n(\mathbf{x}^n) = \frac{\sum_{\mathbf{d}\in s\cdot D} k(s^{-2}|\mathbf{d}|^2)\delta[b(\mathbf{x}^n + \mathbf{d}) - i]}{\sum_{\mathbf{d}\in s\cdot D} k(s^{-2}|\mathbf{d}|^2)} \tag{11}$$

where, $k(x)$ is Epanechnikov kernel profile with support D. After initializing the gradient ascent (at time t) at $\mathbf{y}_0^n = \hat{\mathbf{x}}_{t-1}^n + \mathbf{u}_t^{\mathrm{ssd}}$, the modified algorithm proceeds as follows:

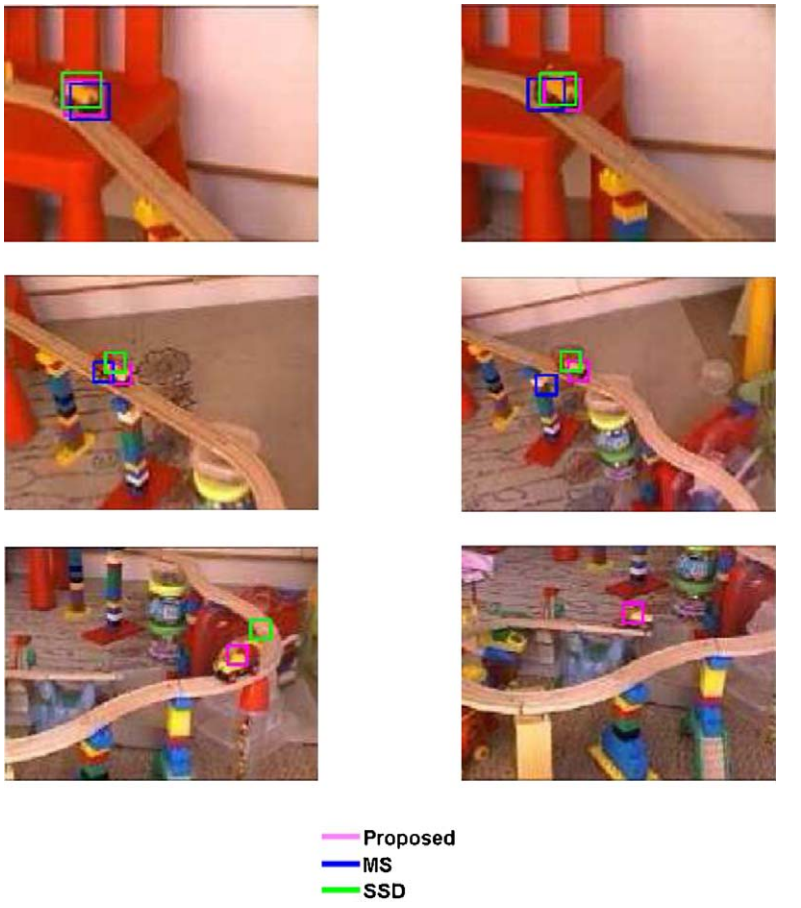

Fig. 6. Tracking results of proposed system (magenta) against SSD (green) and MS (blue) tracker for "train" sequence. Frames shown 20, 50, 200, 270, 620 and 1150.

Please cite this article as: R. Venkatesh Babu et al., Robust tracking with motion estimation and ..., Image and Vision Computing (2006), doi:10.1016/j.imavis.2006.07.016

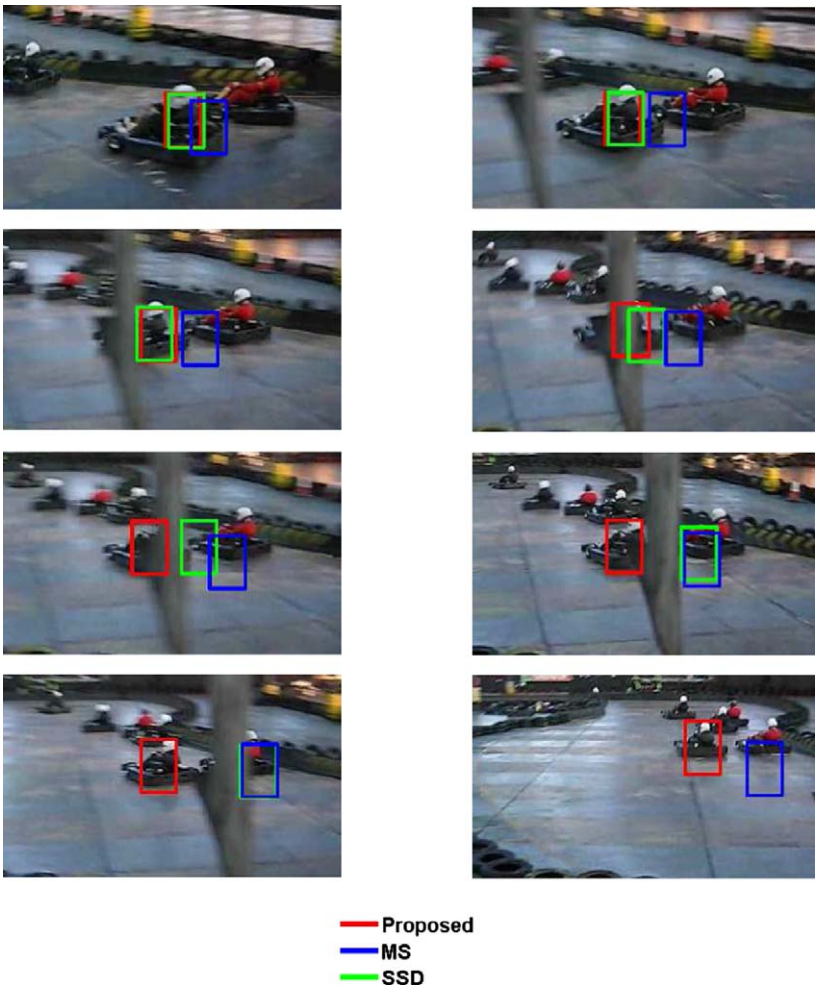

Fig. 7. Tracking result of proposed system (red) against SSD (green) and MS (blue) tracker for "go-carts" sequence. Frames shown 35, 42, 45, 46, 47, 48, 50, and 58.

(1) Given current location $\mathbf{y}_0^n$ compute histogram $\mathbf{p}^n(\mathbf{y}_0^n)$ and Bhattacharya coefficient $\rho(\mathbf{p}^n(\mathbf{y}_0^n), \mathbf{q}^n)$.
(2) Compute candidate position.

$$\mathbf{y}_1^n = \frac{\sum_{\mathbf{d}\in s\cdot D} w^n(\mathbf{y}_0^n + \mathbf{d})k'(s^{-2}|\mathbf{d}|^2)(\mathbf{y}_0^n + \mathbf{d})}{\sum_{\mathbf{d}\in s\cdot D} w^n(\mathbf{y}_0^n + \mathbf{d})k'(s^{-2}|\mathbf{d}|^2)}$$

with weights at location $\mathbf{x}$

$$w^n(\mathbf{x}) = \sum_{i=1}^{m} \sqrt{\frac{q_i^n}{p_i^n(\mathbf{y}_0^n, s)}}\delta[b(\mathbf{x}) - i].$$

(3) while $\rho(\mathbf{p}^n(\mathbf{y}_1^n, s), \mathbf{q}^n) < \rho(\mathbf{p}^n(\mathbf{y}_0^n, s), \mathbf{q}^n)$ do $\mathbf{y}_1^n \leftarrow \frac{1}{2}(\mathbf{y}_1^n + \mathbf{y}_0^n)$.
(4) if $\|\mathbf{y}_1^n - \mathbf{y}_0^n\| < \varepsilon$ the stop, otherwise set $\mathbf{y}_0^n \leftarrow \mathbf{y}_1^n$ and repeat Step 2.
(5) Use only the reliable displacements out of N measurements for the final estimate. Let $\mathbf{R} \subset \{\mathbf{y}^1 \ldots \mathbf{y}^N\}$ be the set of all reliable MS trackers. The final global motion estimate is obtained as: $\mathbf{y} = mean(\mathbf{y}^i)$, $i \in \mathbf{R}$.

The final estimate provides the displacement estimate $\mathbf{u}_t = \mathbf{y} - \hat{\mathbf{x}}_{t-1}$. In our experiments, the MS trackers whose Bhattacharya coefficients lie in the top 10 percent are considered as reliable MS trackers. Finally, the two entries associated with this measurement in the covariance matrix R_t of the observation model (2) are chosen as

$$\sigma_u^2 = \sigma_v^2 = \mathrm{e}^{\tau(1-\mathrm{mean}\{\rho_i\})}, \quad i \in \mathbf{R} \tag{12}$$

where, ρ_i are the Bhattacharya coefficients of the reliable MS trackers. The parameter τ is set to 25 in the experiments.

3.3. Model update

The presented video sequences were shot with a hand-held camcorder, which automatically adjusts the brightness

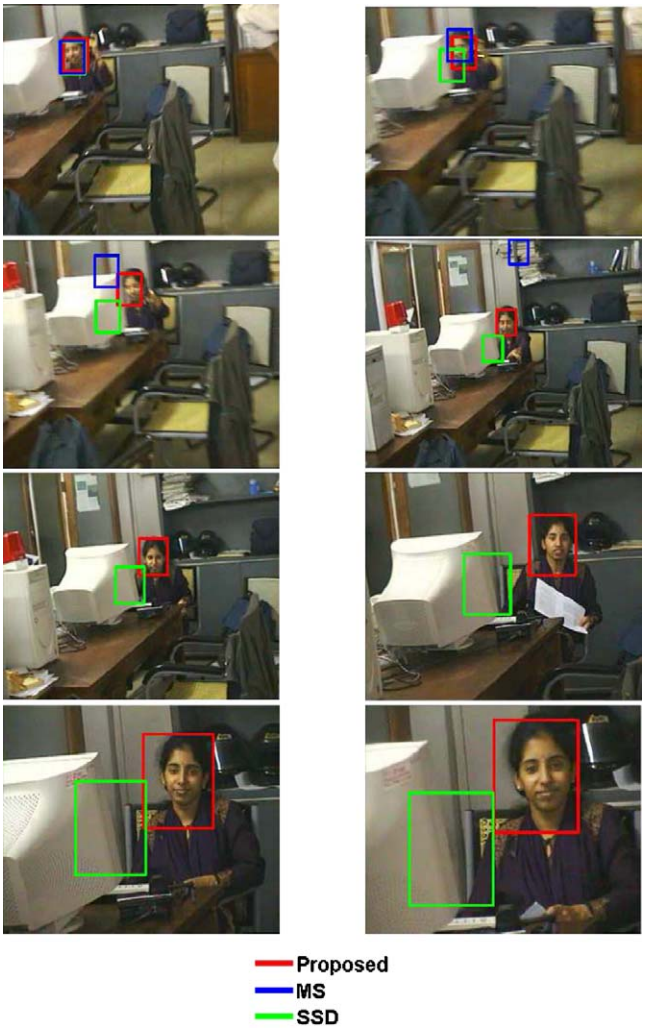

Fig. 8. Tracking result of proposed system (red) against SSD (green) and MS (blue) tracker for "lab" sequence. Frames shown 53, 56, 59, 101, 149, 200, 251, and 296.

based on the background illumination. The change of object color in these videos is due to this automatic adjustment of camera parameters. In such videos, working with a fixed color model would drastically reduce tracking accuracy. The target in sequence shown in Fig. 11 not only undergoes partial occlusion, but also object luminance changes drastically from the start to the final frame. In such cases, it is necessary to adapt the tracking model to brightness/color change. In our system, the tracking points whose support lies mostly within the object are updated with the latest color model. The area of intersection between the object and the support of each tracking point is estimated using the current log-likelihood map. Fig. 5 shows the log-likelihood map when the object undergoes partial occlusion. The model corresponding to a particular tracking point is updated if the object area occupies a certain minimal fractional area (in our system it is set as 50%) of its support region. Only those support regions of the tracking points that intersect with the object are used for updating the target color model of the corresponding tracking points. Let M_t be the binary mask obtained from the log-likelihood map:

$$M_t(i) = \begin{cases} 1 & \text{if} L_t(i) > th_u \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

In our experiments, threshold th_u is set as 2 for considering only the most reliable object pixels. Let $\mathbf{q}^n$ be one of the MS models located at $\mathbf{x}$ with support region $s \cdot D + \mathbf{x}$. If $\frac{1}{|D|}\sum_{d\in D} M_t(s \cdot D + \mathbf{x}) > D_{th}$, then replace the model $\mathbf{q}^n$ with the recent model obtained as:

$$q_i^n(\mathbf{x}) = \frac{\sum_{\mathbf{d}\in D} M_t(s \cdot D + \mathbf{x}) k(s^{-2}|\mathbf{d}|^2) \delta[b(\mathbf{x}+\mathbf{d}) - i]}{\sum_{\mathbf{d}\in D} M_t(s \cdot D + \mathbf{x}) k(s^{-2}|\mathbf{d}|^2)} \tag{14}$$

In our experiments, D_{th} is set as 0.5 (corresponding to 50% of the support area).

3.4. Algorithm summary

The complete algorithm is summarized below. Given previous reference color models $\mathbf{q}_{t-1}^n$ and previous state estimate $(\hat{\mathbf{x}}_{t-1}, \hat{s}_{t-1})$ with error covariance R_{t-1}:

(1) Obtain the thresholded likelihood map T_t of the object/background according to Eq. (9)
(2) Obtain SSD-based displacement measurement $\mathbf{u}_t^{\text{ssd}}$ according to Eq. (10) with the weighting factor (T_t).
(3) Correct this measurement with reliable MS trackers, initialized at $\mathbf{u}_t^{\text{ssd}}$ and with reference color models $\mathbf{q}_{t-1}^n$, to obtain final measurement $\mathbf{u}_t$.
(4) Update the target models of reliable MS tracking modules.
(5) Estimate global affine motion over the image and derive new scale measurement ξ_t according to Eq. (6).
(6) Using displacement and scale measurement $\mathbf{u}_t$ and ξ_t, update state estimate with Kalman filter, providing $(\hat{\mathbf{x}}_t, \hat{s}_t)$ and associated error covariance P_t.

Initial state $(\hat{\mathbf{x}}_1, \hat{s}_1 = 1)$ in frame 1 (and associated support region) is obtained either by manual interaction or by detection, depending on the scenario of interest.

4. Results and discussion

The computational complexity of the proposed tracker is the combined (sum) complexities of the individual SSD and MS trackers. As known, the complexity of MS tracker is suitable for real-time tracking. Hence, it is the SSD tracker that critically determines the speed of the proposed algorithm. The complexity of SSD tracker is proportional to the size of the search range as well as the object being tracked. Most objects typically occupy a small region in the whole frame. Besides, the search range is usually limited to a small window around the object. Exploiting the two characteristics, SSD tracker

Please cite this article as: R. Venkatesh Babu et al., Robust tracking with motion estimation and ..., Image and Vision Computing (2006), doi:10.1016/j.imavis.2006.07.016

Fig. 9. Tracking result of proposed system against the [SSD+global MS] on "walk" sequence for frames 2, 20, 40, 100, 140, and 200 are shown. The '+' marks indicate the MS tracking points, the red rectangle corresponds to the proposed tracking and the green rectangle corresponds to the [SSD + global MS] tracker result.

can be used for real-time tracking of objects as illustrated by Hager et al. [31]. Our proposition of initialization of object location by SSD tracker, brings down the number of mean-shift iterations, since mean-shift iterations are only used here for finer adjustments of object location. These aforementioned factors work in favor of speeding the algorithm, making it feasible for real-time applications. Since the objective of the SSD tracker is to provide the approximate initial location for MS tracker, it can also be replaced with some of the well known fast-search techniques [32–34].

In the following subsections, we discuss the experimental results for the proposed combined tracker, as well as its performance with the addition of the module for occlusion handling.

4.1. Performance of the combined tracker

The proposed algorithm has been tested on several complex video sequences. Most of the videos used in our experiments are shot by a hand-held camcorder containing a wide variety of camera operations. The presented videos were chosen for the camera operations they include. For example the "train" sequence contains lots of jerky motion with camera pan, tilt and zoom operations. In the "go-carts" sequence, the fast moving go-carts move away from the camera, creating shrinking effect of the object. This sequence contains pan tilt camera operations also. In both "train" and "go-carts" sequence the objects undergo significant appearance changes too. The "lab" sequence contains

Please cite this article as: R. Venkatesh Babu et al., Robust tracking with motion estimation and ..., Image and Vision Computing (2006), doi:10.1016/j.imavis.2006.07.016

Fig. 10. Tracking result of proposed system on a movie sequence "run-lola-run" is shown. The '+' marks indicate the MS tracking points, the red rectangle corresponds to the proposed tracking result.

camera pan, tilt and zoom operations. The experimental results show that the proposed tracking system, which uses both SSD and MS tracker, works better than either of them used individually. Tracking results for these personal videos of low quality are presented in Figs. 6–8.

The first sequence includes jerky movements that make it challenging for the trackers to follow the objects. However, the proposed algorithm was able to track the toy train throughout the sequence. It is observed that the mean-shift tracker oscillates about the object whenever an intense jerk occurs, and gets lost later. On the other hand, the SSD tracker performs well till about the 600th frame and collapses after an intense jerk. It can be seen that the toy train turns around almost 180 degrees from starting to end, and experiences substantial scale changes due to camera zooming in and out. The model update helps here to learn the object while tracking.

In the second sequence, fast camera movements and those of racing go-carts result in large displacements in the image and dramatic motion blur. In addition, go-carts get briefly occluded. Despite these difficulties, the combined tracker manages to successfully track the target go-cart, whereas, when the SSD and MS trackers perform individually, they lose track of the target.

For the third sequence, results were presented for frames that are temporally sub-sampled by 3. In this sequence, rapid camera movements make the MS and SSD tracker fail, while the combined tracker faithfully tracks the object, efficiently handling the consequences of zooming.

4.2. Performance of the proposed tracker with occlusion handling

The proposed algorithm has been tested on several videos and it is observed to have performed well, not only under partial, but also a brief span of complete occlusion. The tracking result for "walk" sequence is shown in Fig. 9 for both, proposed tracker as well as the [SSD + global MS] tracker. In this sequence, the object undergoes partial occlusion. The proposed system was able to track the object correctly without any shift even when the object got partly occluded. The global MS-based tracker vastly deviates from the desired result during partial occlusion. Tracking result of the proposed algorithm for a dynamic video shot from the movie "Run, Lola, run" is shown in Fig. 10. The number of MS tracking points used in "walk" sequences were 20, while 15 MS tracking points were used for "run-lola-run" sequence.

The results obtained with such update model is shown in Fig. 11. In this example, the [SSD + global MS] tracker does not perform satisfactorily. The proposed method without model update tracks the object till the end of sequence but suffers deviation from the object due to luminance/color change of the object. However, the proposed tracker with model update is able to track the object faithfully, with no deviation.

5. Conclusion

In this paper, we have proposed an efficient visual tracker by coupling SSD and mean-shift algorithms, which have

Please cite this article as: R. Venkatesh Babu et al., Robust tracking with motion estimation and ..., Image and Vision Computing (2006), doi:10.1016/j.imavis.2006.07.016

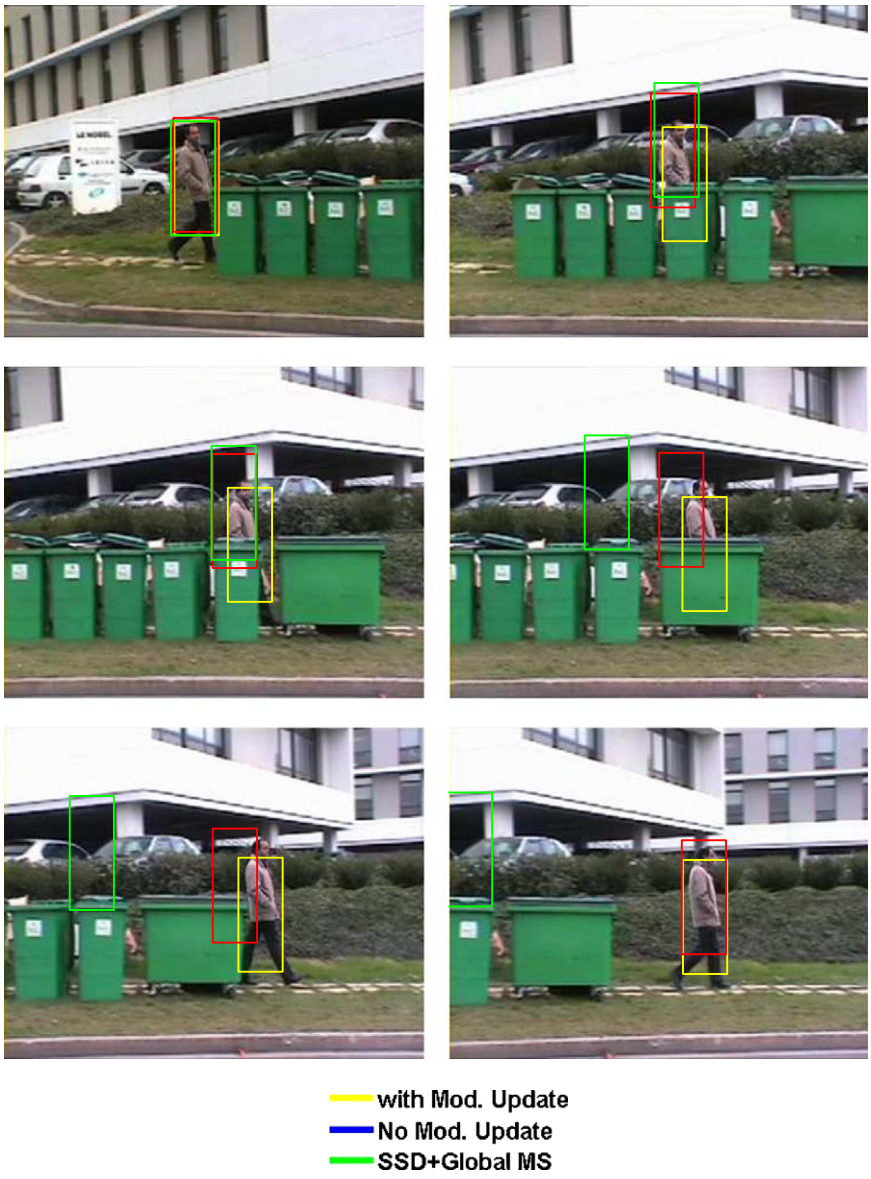

Fig. 11. Tracking result of proposed system against the [SSD + global MS] tracker on another "walk" sequence is shown. The yellow rectangle corresponds to to the proposed tracking with model update, the red rectangle corresponds to the proposed tracking without model update and the green rectangle corresponds to the [SSD + global MS] tracker result.

complementary properties. Further, tracking local color properties of the object using multiple MS tracking points on the object, instead of a single global MS tracker, improves the performance when the object undergoes partial occlusion. The improved performance of the proposed tracker, over combined SSD and global mean-shift tracker, is proved using various video sequences. Since both trackers have real-time computational complexity, the proposed compound tracker is suitable for real time tracking of objects.

References

[1] B. Lucas, T. Kanade, An iterative image registration technique with an application to stereo vision, in: International Joint Conference on Artificial Intelligence, 1981, pp. 674–679.

[2] G. Hager, P. Belhumeur, Efficient region tracking with parametric models of geometry and illumination, IEEE Trans. Pattern Anal. Mach. Intell. 20 (10) (1998) 1025–1039.

[3] M. Isard, A. Blake, Contour tracking by stochastic propagation of conditional density, in: European Conference on Computer Vision, 1996, pp. 343–356.

[4] P. Fieguth, D. Terzopoulos, Color based tracking of heads and other mobile objects at video frame rates, in: IEEE Conference on Computer Vision Pattern Recognition, 1997, pp. 21–27.

[5] D. Comaniciu, V. Ramesh, P. Meer, Kernel-based object tracking, IEEE Trans. Pattern Anal. Mach. Intell. 25 (5) (2003) 564–577.

[6] J. Yang, A. Waibel, A real-time face tracker, in: WACV, 1996, pp. 142–147.

[7] C.R. Wren, A. Azarbayejani, T. Darrell, A. Pentland, Pfinder: real-time tracking of the human body, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 780–785.

[8] G. Bradski, Computer vision face tracking as a component of a perceptual user interface, in: Workshop on Application of Computer Vision, Princeton, NJ, 1998, pp. 214–219.

Please cite this article as: R. Venkatesh Babu et al., Robust tracking with motion estimation and ..., Image and Vision Computing (2006), doi:10.1016/j.imavis.2006.07.016

[9] G. Cheung, S. Baker, T. Kanade, Shape-from silhouette of articulated objects and its use for human body kinematics estimation and motion capture, in: IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, 2003, pp. 77–84.
[10] M.J. Black, A.D. Jepson, Eigentracking: robust matching and tracking of articulated objects using a view-based representation, Int. J. Comput. Vis. 26 (1) (1998) 63–84.
[11] S. Avidan, Support vector tracking, in: IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, 2001, pp. 184–191.
[12] O. Williams, A. Blake, R. Cipolla, Sparse bayesian learning for efficient visual tracking, IEEE Trans. Pattern Anal. Mach. Intell. 27 (8) (2005) 1292–1304.
[13] D. Comaniciu, V. Ramesh, P. Meer, Real-time tracking of non-rigid objects using mean shift, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head, SC, 2000.
[14] A. Jepson, D. Fleet, T. El-Maraghi, Robust online appearance models for visual tracking, IEEE Trans. Pattern Anal. Mach. Intell. 25 (10) (2003) 1296–1310.
[15] H. Nguyen, A. Smeulders, Fast occluded object tracking by a robust appearance filter, IEEE Trans. Pattern Anal. Mach. Intell. 26 (8) (2004) 1099–1104.
[16] K. Fukunaga, L.D. Hostetler, The estimation of the gradient of a density function, with applications in pattern recognition, IEEE Trans. Inform. Theo. 21 (1) (1975) 32–40.
[17] Y. Cheng, Mean shift, mode seeking, and clustering, IEEE Trans. Pattern Anal. Mach. Intell. 17 (8) (1995) 790–799.
[18] D. Comaniciu, P. Meer, Mean shift: a robust approach toward feature space analysis, IEEE Trans. Pattern Anal. Mach. Intell. 24 (5) (2002) 603–619.
[19] F. Porikli, O. Tuzel, Multi-kernel object tracking, IEEE Int. Conf. Multimedia Expo (2005) 1234–1237.
[20] C. Shen, M.J. Brooks, A. van den Hengel, Fast global kernel density mode seeking with application to localisation and tracking, in: IEEE International Conference on Computer Vision, vol. 2, 2005, pp. 1516–1523.
[21] C. Yang, R. Duraiswami, L. Davis, Efficient mean-shift tracking via a new similarity measure, in: IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, 2005, pp. 176–183.
[22] C. Shan, Y. Wei, T. Tan, F. Ojardias, Real time hand tracking by combining particle filtering and mean shift, in: IEEE International Conference on Automatic Face and Gesture Recognition, 2004, pp. 669–674.
[23] G. Hager, M. Dewan, C. Stewart, Multiple kernel tracking with SSD, in: IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, 2004, pp. 790–797.
[24] K. Deguchi, O. Kawanaka, T. Okatani, Object tracking by the mean-shift of regional color distribution combined with the particle-filter algorithms, in: International Conference on Pattern Recognition, vol. 3, 2004, pp. 506–509.
[25] Z. Zhu, Q. Ji, K. Fujimura, Combining kalman filtering and mean shift for real time eye tracking under active IR illumination, in: Proceedings of International Conference on Pattern Recognition, Quebec City, Canada, vol. 4, 2002, pp. 318–321.
[26] P. Pérez, C. Hue, J. Vermaak, M. Gangnet, Color-based probabilistic tracking, in: Proceedings of European Conference on Computer Vision, Copenhagen, Denmark, 2002.
[27] K. Nummiaro, E. Koller-Meier, L. Van Gool, An adaptive color-based particle filter, Image Vis. Comput. 21 (1) (2003) 99–110.
[28] R. Collins, Mean-shift blob tracking through scale space, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Madison, Wisconsin, vol. 2, 2003, pp. 234–241.
[29] Z. Zivkovic, B. Kröse, An EM-like algorithm for color-histogram-based object tracking, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC, vol. 1, 2004, pp. 798–803.
[30] J.-M. Odobez, P. Bouthemy, Robust multiresolution estimation of parametric motion models, J. Vis. Commun. Image R. 6 (4) (1995) 348–365.
[31] G. Hager, P.N. Belhumeur, Real-time tracking of image regions with changes in geometry and illumination, in: IEEE Conference on Computer Vision and Pattern Recognition, 1996, pp. 403–410.
[32] R. Li, B. Zeng, M. Liou, A new three-step search algorithm for block motion estimation, IEEE Trans. Circ. Syst. Video Technol. 4 (4) (1994) 438–441.
[33] L.-M. Po, W.-C. Ma, A novel four-step search algorithm for fast block motion estimation, IEEE Trans. Circ. Syst. Video Technol. 6 (3) (1996) 313–317.
[34] S. Zhu, K.-K. Ma, A new diamond search algorithm for fast block-matching motion estimation, IEEE Trans. Image Process. 9 (2) (2000) 287–290.

Please cite this article as: R. Venkatesh Babu et al., Robust tracking with motion estimation and ..., Image and Vision Computing (2006), doi:10.1016/j.imavis.2006.07.016