

Improved Tracking of Multiple Humans with Trajectory Prediction and Occlusion Modeling

Rómer Rosales and Stan Sclaroff
Image and Video Computing Group
Computer Science Department
Boston University
Boston, MA 02215

Abstract

A combined 2D, 3D approach is presented that allows for robust tracking of moving bodies in a given environment as observed via a single, uncalibrated video camera. Low-level features are often insufficient for detection, segmentation, and tracking of non-rigid moving objects. Therefore, an improved mechanism is proposed that combines low-level (image processing) and mid-level (recursive trajectory estimation) information obtained during the tracking process. The resulting system can segment and maintain the tracking of moving objects before, during, and after occlusion. At each frame, the system also extracts a stabilized coordinate frame of the moving objects. This stabilized frame can be used as input to motion recognition modules. The approach enables robust tracking without constraining the system to know the shape of the objects being tracked beforehand; although, some assumptions are made about the characteristics of the shape of the objects, and how they evolve with time. Experiments in tracking moving people are described.

1 Introduction

Tracking non-rigid objects and classifying their motion is a challenging problem. Many of the key obstacles are not solved yet. The importance of tracking and motion recognition problems is evidenced by the increasing attention they have received in recent years [30]. Effective solutions to these problems would lead to breakthroughs in areas such as video surveillance, motion analysis, virtual reality interfaces, robot navigation and recognition.

Low-level image processing methods have been shown to work surprisingly well in restricted domains despite the lack of high-level models; *e.g.*, [11, 10, 21, 34, 9]. Unfortunately, most of these techniques assume a simplified version of the general problem; *e.g.*, there is only one moving object, objects do not occlude each other, or objects appear at a limited range of scales and orientations. While higher-level, model-based techniques can address some of these problems, such methods require guidance by a human operator, usually for model placement or initialization.

The main contributions of this work are: 1.) to extend the low-level techniques to handle multiple moving objects, 2.) to explicitly model occlusion, and 3.) to estimate and predict 3D motion trajectories. Because simply using low-level features for detection, segmentation, and tracking of moving, non-rigid objects is often insufficient, an

improved mechanism is proposed that combines low-level (image segmentation) and mid-level (recursive trajectory estimation) information obtained during the tracking process. This approach enables robust tracking without constraining the system to know the shape of the objects being tracked beforehand, although some assumptions are made about the characteristics of the shape of the objects, and how they evolve with time.

Occlusion cannot be ignored in the segmentation of multiple moving objects. We therefore provide an approach to detect and to maintain the tracking of moving objects before, during, and after an occlusion. This approach enables accurate extraction of a stabilized coordinate frame for moving non-rigid objects that can be useful for further motion analysis. Highly articulated human bodies are represented in two different ways: 1.) a region of support, and 2.) a 2D bounding box that bounds the body.

Our work will focus on tracking using video information provided by a single camera located in the same local area where the activities occur; *e.g.*, in a living room, or on a street corner. This is in contrast to approaches that assume a top or very distant view of the scene, because it provides more substantial information for higher level mechanisms to be developed; *e.g.*, human motion analysis and understanding, recognition, and virtual reality interaction. Many techniques developed for these purposes have had the problem that registration of useful, filtered information is a hard labor by itself [11, 23, 25, 35]. From these points of view, our system can be a functional front-end that would support these tasks.

2 Related Work

One of the fundamental ideas in motion perception is the work of Johansson's moving light displays [17], where it was demonstrated that relatively little information is needed (in theory) to perform motion recognition. Perhaps one of the first approaches related with walking people in real environments is due to Hogg [14].

The basic detection and registration technique used in our approach, based on background segmentation, is related to the work of Baumberg and Hogg [3] and Bichsel [5]. Using similar differencing techniques are [34, 2], whose likelihood measurement approach is closely related to ours; however, they used it in the simpler problem of detecting a single moving object. Nonetheless, these works showed how this information can be used as input for more

sophisticated techniques.

These detection and registration methods have also been tested as the basis of more complicated representations like [19, 13, 24, 29, 18, 27], who used model-based techniques, generally articulated models comprised of 2D or 3D solid primitives. Others have used multiple but rigid structures. For instance [20], who used it for tracking cars. [15] uses it for tracking multiple objects, in which the projective effect is avoided through the use of a top view. Most of the techniques mentioned have not been tested in areas where multiple non-rigid objects interact and/or the object paths undergo the projective effect of image formation.

Rigidity of objects has been explicitly exploited by [20, 12, 8, 4]. Our approach is related more to these because in some situations we use the idea of a global rigid body when fixing the feature points.

For estimating motion trajectories, we use the Extended Kalman Filter (EKF). For a good review, see [31]. The EKF has been extensively used in computer vision [8, 28, 1, 22, 26, 6, 9]. Another important set of approaches is based on [16]. Our formulation has some similarities to [1].

3 Approach

An overview of our approach is illustrated in Fig. 1. The first stage of the algorithm is based on the background subtraction methods of [34]. The system is initialized by acquiring statistical measurements of the empty scene over a number of video frames. The statistics acquired are the mean and covariance of image pixels in 3D color space. The idea is to have a confidence map to guide segmentation of the foreground from the background.

Once initialized, moving objects in the scene are segmented using a log-likelihood measure between the incoming frames and the current background model. The input video stream is low-pass filtered to reduce noise effects.

A connected components analysis is then applied to the resulting image. Initial segmentation is usually noisy, so morphological operations and size filters are applied. The output of this segmentation process is shown in the second set of images in Fig. 2.

If there are occlusions, or strong similarities between the empty scene model and the objects, then it is possible that regions belonging to the same object may be classified as part of different objects or vice versa. To solve this problem, two sources of information are used: temporal analysis and trajectory prediction.

In temporal analysis, a map of the previous segmented and processed frame is kept. This map is used as a possible approximation of the current connected elements. Connectivity in the current frame is compared with it, and regions are merged if they were part of the same object in previous frames. This can account for some temporal shadow effects, local background foreground similarities and short occlusions.

In trajectory prediction, an Extended Kalman Filter

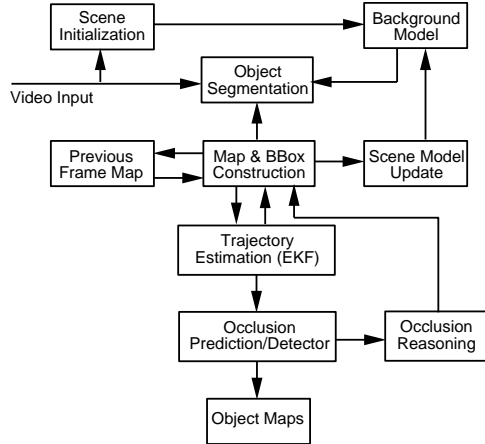


Figure 1: System Diagram.

(EKF) provides an estimate of each object's position and velocity. If noisy observations are perceived, the EKF provides a natural way to deal with them. For example, if part of one object is not observed in a consecutive number of frames, an approximation of the predicted bounding box of the object is used. During an occlusion, the EKF can be used to give the maximum likelihood estimate of the current region covered by the object, along with its velocity and position. The estimated bounding box prediction for each person is shown in the bottom row of Fig. 2.

Occlusion prediction is performed based on the current estimations of the filter. Given that we know object position and the occupancy map, we can detect occlusions or collisions in the image plane. Our EKF formulation estimates the trajectory parameters for these objects assuming locally linear 3D motion, also the bounding box parameters are estimated.

During an occlusion, observations are used to limit the output of the filter, but single object position and silhouette cannot be observed. The end of an occlusion event can also be detected. When this happens, our confidence in the observation changes our estimation process to the initial state (no occlusions). Every aspect is discussed in more detail in the following sections.

As shown in Fig. 3, a stabilized coordinate frame of the object is easily constructed with the information extracted from the sequence. The moving object is resampled in a canonical frame throughout the tracking sequence, despite changes in scale and position. The estimated bounding box is used to resize and resample the moving blob into a canonical view that can be used as input to motion recognition modules.

We are using human body tracking as a particular application of our formulation; however, human body structure and articulation characteristics may not be the best suited example for our assumptions. This is mainly due to its non-planar structure and large number of degrees of freedom. This creates an enormous set of configurations in general.

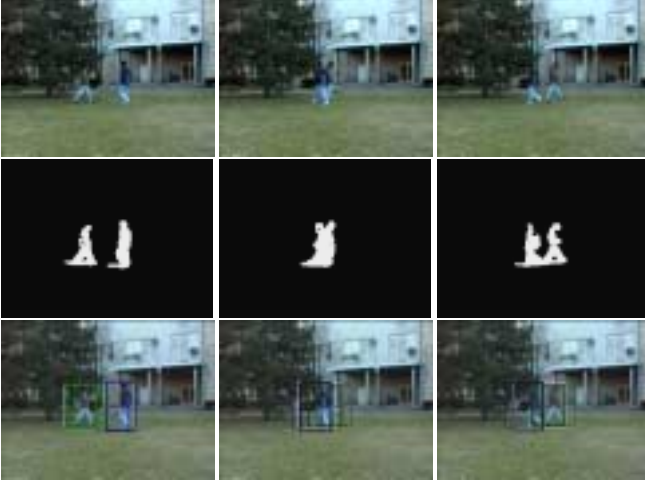


Figure 2: Tracking before, during, and after occlusion. The top row shows frames from the input sequence of two people walking across a grass courtyard. The next row shows the connected components extracted from the background. The bottom row shows the corresponding estimates of the bounding boxes for the moving objects. The bounding boxes are successfully predicted during the occlusion via an EKF.

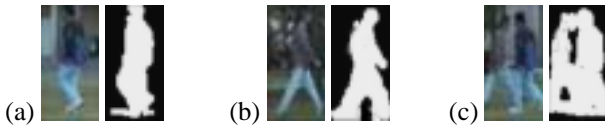


Figure 3: Example pairs of normalized views (support map and real view). The moving object is resampled in a canonical frame throughout the sequence, despite changes in scale and position.

But one advantage of not considering specific details about structure and constraints about human motion is that our approach can be applied to almost any object.

4 Registration of Multiple Objects

In order to detect object motion or location to track, an accurate registration technique is needed. The use of a fixed camera makes background-foreground segmentation methods well suited for this task [34, 3, 5, 15, 29]. If controlled with a feedback approach, such methods can achieve a good level of accuracy [34]. More importantly, this approach is efficient enough to allow real-time tracking. Finally, these techniques also allow detection of foreground objects even when they are not moving.

Another possible approach is to measure the optical flow and use it to guide registration and/or segmentation. Unfortunately, non-rigid body tracking cannot easily benefit from optical flow computation, mainly because it relies too heavily on the constant brightness assumption (which obviously is not a good assumption in our case). Also the complicated vector patterns that are generated due to the motion of the different parts of the given body will not give

us more than what we can obtain from the previous methods without more expensive computations. Occlusions due to multiple moving objects further complicate this process.

4.1 Scene Initialization

Our two initial steps are based on the results of [34]. We first build a representation of the background. This is done by capturing the environment without interesting objects (those we want to track). For the empty scene to be learned, it is necessary to capture about 2 seconds (approx. 60 frames) of video. The idea is to obtain a good estimate of the color covariance at each image pixel.

Second order statistics are then estimated. For a given pixel, we estimate the mean $\hat{\mu}$ in 3D color space. The covariance for the color distribution of that pixel is then estimated using:

$$K_{xx} = E[(x - \hat{\mu}_x)(x - \hat{\mu}_x)^T], \quad (1)$$

where x is the observed vector in color space (corresponding to a given the pixel). The initial scene model is given by the pair $(\hat{\mu}, K)$, and will be updated as described below.

Using this, we learn the model of our background. The information obtained in this process will make the segmentation more robust to sensor noise, and to changes in small areas of the scene that present some kind of movement. Examples of this are leaves that move due to the action of wind, some specularities, etc.

4.2 Segmentation

We will first describe the segmentation procedure used during regular conditions (no occlusions). Objects in the scene are characterized by a relatively large variation of the background statistics. Changes in the learned scene are computed at every frame by comparing current frames with the model. For this a distance measure is computed between the incoming image and the background using the log likelihood measure:

$$\mathbf{d}_x(t) = -(\mathbf{x}(t) - \hat{\mu}(t))^T K_{xx}^{-1} (\mathbf{x}(t) - \hat{\mu}(t)) - \ln |K_{xx}^{-1}| \quad (2)$$

This provides an estimate of the similarity with the background at every pixel. Then if a region of the image is dissimilar enough to the estimated empty region, the system marks it as an interesting region.

A binary map of the current frame is computed with the interesting regions in it. Then basic image morphological operations are applied (erosion-dilation) in order to rule out small regions and to fill holes in probable regions of interest. The binary map is represented by:

$$\mathbf{b}_i(t) = \bigcup_x [\mathbf{d}_x(t) > \Gamma] \quad (3)$$

The connected region $\mathbf{b}_i(t)$ at time t is defined by the locations where there was a difference from the model

greater than a given threshold Γ . For this initial map, connected regions are labeled using a connected component analysis algorithm and their bounding boxes are computed. Initially different connected regions are merged [15]. As opposed to previous approaches, our method includes two additional steps. First, bounding boxes are tested for collision and merged if so. Second, information from the object segmentation obtained in the previous frame is used. By assuming that object trajectories are continuous and that their position changes at a rate smaller than their own size in the direction of motion every 1/30th of a second, a temporal matching is done. When occlusions are suspected or detected this process changes as will be described in Sec. 6.

At the end of this process a presumable object labeling is obtained where blobs are separated to represent the different moving objects thought to be present in the scene. One possible drawback of this is the case when objects appear in the scene for the first time being too close to each other. Because we are not using an explicit or hand-crafted model of the human body shape, this will be labeled to be one single object. The problem is different when objects collide after being present in the scene as different objects, in this case our occlusion reasoning process will try to identify and correctly label them.

In conclusion, we denote the final segmentation as $s_j(t) = g(\mathbf{b}(t))$, where $g(\bullet)$ uses all blobs found in our initial step. Our scene map at time t is $\mathbf{M}_t = \bigcup_j [s_j(t)]$.

4.3 Updating the Model

After a binary map \mathbf{M}_t is obtained with all the possible objects in the scene, the system updates the statistical properties of the scene background, in this implementation only the mean is updated, this is done using a simple adaptive filter:

$$\hat{\mu}(t) = \alpha \mathbf{X}(t-1) \mathbf{M}_{t-1} + (1-\alpha) \hat{\mu}(t-1) \overline{\mathbf{M}}_{t-1}, \quad (4)$$

where $\overline{\mathbf{M}}(t)$ is the complement of $\mathbf{M}(t)$ at time t , $\mathbf{X}(t)$ is the input frame at time t , α is a gain constant.

This can account for slow changes in lighting conditions in the scene, and will keep our scene model updated in case new objects of no apparent interest are introduced.

4.4 Tracker Structure

A *tracker* object T , is assigned to every object being tracked. It contains information about object location, a binary support map, blob characteristics, bounding box, and other information that is going to be used for tracking using the EKF formulation explained in Sec. 5.

A new tracker structure is assigned to every detected object. Special care needs to be taken with new objects in order to produce a stable tracker. There is a problem when objects appear within the boundary of other objects already present. In such cases the system may misclassify the objects as being the same. But in the general case, new objects can be detected easily by keeping a global state of

objects being tracked. These states are maintained by our tracker instances T_i .

5 Trajectory Estimation

An extended Kalman filter (EKF) is used to recursively predict the objects' future positions based on their current positions and velocities. The parameterization and selection of the state vector is in part related to the work of [1], who used it in a structure from motion problem. However, they tested their approach with rigid structures with the assumption that enough number of feature points are tracked during the sequence. The problem of feature correspondence and tracking was not addressed directly. Furthermore, their method could not handle the appearance (or disappearance) of new features. Our approach is also related to the ideas of [8] in the use of an EKF to predict and recover object trajectories.

5.1 Features

To reduce the complexity of the tracking problem, two feature points are selected: two opposite corners in the object bounding box. By using this we avoid the problem of searching for features and establishing correspondence between them in consecutive frames. In general we think that a detailed tracking of features is neither necessary nor easily tenable for non-rigid motion tracking. In fact, it may even be impossible without a detailed model of the object being tracked.

It is therefore assumed that although the object to be tracked is highly non-rigid, the 3D size of the object's bounding box will remain approximately the same, or at least vary smoothly. This assumption might be too strong in some cases; *e.g.*, if the internal motion of the object's parts cannot be roughly self contained in a bounding box. However, when analyzing basic human locomotion, we believe that these assumptions are a fair approximation.

5.2 Camera Model

For our representation a 3D central projection model similar to [32, 1] is used:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \frac{1}{1+z\beta}, \quad (5)$$

where (x, y, z) is the real 3D feature location in the camera reference frame, (u, v) is the projection of it to the camera plane, and $\beta = \frac{1}{f}$ is the inverse focal length. The origin of the coordinate system is fixed at the image plane. This model has proven to be useful when estimating focal length and structure in the structure from motion problem [1]. One important property of this model is that it is numerically well defined even in the case of orthographic projection.

5.3 3D Representation of Feature Points

One of the novel techniques here introduced is the use of a convenient parameterization of the feature points. The

main purpose is to extract motion information and use it as a set of constraints to estimate the relative 3D position of the features. By assuming that the objects are undergoing a locally linear translational motion in 3D, we can extract relative 3D position of the object (or features) using several frames.

Feature point location will be parameterized in terms of the object's x, y coordinates and its depth z coupled with β . Similarly, we use the vector $vel = (\dot{x}, \dot{y}, z\dot{\beta})$ to represent a feature's 3D velocity relative to the camera reference frame. This formulation only allows for recovery of depth up to a scale factor. For our tracking application, there is no need to recover the true 3D position of the feature point.

The noise sensitivity of a similar formulation has been analyzed in [1]. Without the effect of registration inaccuracies, the sensitivity in \dot{x} and \dot{y} is directly dependent on the object depth. This is because objects that are farther away from the camera tend to project to fewer image pixels. The sensitivity of \dot{z} depends on camera focal length. As focal length increases, sensitivity decreases. In the orthographic case this sensitivity is zero. In general, the sensitivity to \dot{x} and \dot{y} is higher than to \dot{z} .

5.4 Extended Kalman Filter Formulation

We will use a first order EKF. We also ran some experiments with a Discrete Kalman Filter (DKF); however, it became clear that the underlying process is inherently non-linear and therefore an EKF is better suited. Our state models a planar rectangle moving along a linear trajectory at constant velocity. Because we are considering the objects as being planar, the depth at both feature points should be the same. The reduction in the number of degrees of freedom improves the speed of convergence of the EKF. Our state vector then becomes:

$$\mathbf{x} = (x_0, y_0, x_1, y_1, z\beta, \dot{x}_0, \dot{y}_0, \dot{x}_1, \dot{y}_1, z\dot{\beta}). \quad (6)$$

Note that because this formulation does not provide a unique solution in 3D space, a scale factor has to be set. However, the family of allowable solutions all project to a unique solution on the image plane. We can therefore estimate objects' future positions on the image plane given their motion in $(x, y, z\beta)$ space.

The measurement relationship to the process is nonlinear, so an EKF formulation is used which linearizes about the current mean and covariance. In other words, we linearize around our current estimate using the measurement partial derivatives. As will be seen in our experiments, motions that are not linear in 3D can also be tracked, but the estimate at the locations of sudden change in velocity or direction is more prone to error. The speed of convergence to new values depends on the filter's expected noise.

In order to detect when the EKF does not represent the true observations we use a simple resetting mechanism that compares the current estimate with the observation. This can be achieved by transforming the state estimate into the observation space using $h(\bullet)$ as described below.

We now briefly explain the EKF formulation. Our process is guided by the following linear difference equation:

$$x_{k+1} = A_k x_k + w_k, \quad (7)$$

where x_k is our state at time k , w_k is the process noise and A_k , the system evolution matrix, is based on first order Newtonian dynamics and assumed time invariant ($A_k = A$). If additional prior information on dynamics is available, then A can be changed to better describe the system evolution [28]. In our case, we use the assumption that trajectories are locally linear in 3D.

Our measurement vector is $z_k = (u_{0k}, v_{0k}, u_{1k}, v_{1k})$, where u_{ik}, v_{ik} are the image plane coordinates for the observed feature i at time k . The measurement vector is related to the state vector via the measurement equation: $z_k = h(x_k + v_k)$. Note that $h(\bullet)$ is non-linear, as described in Sec. 5.2. Measurement noise is assumed to be additive in our model. The EKF time update equation becomes:

$$\hat{x}_{k+1} = A_k \hat{x}_k \quad (8)$$

$$P_{k+1}^- = A_k P_k A_k^T + W Q_k W^T \quad (9)$$

where A represent the system evolution transformation, Q_k is the process noise covariance. The matrix W is the Jacobian of the transformation A with respect to w .

Finally, the measurement update equations become:

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V R_k V^T)^{-1} \quad (10)$$

$$\hat{x}_k = \hat{x}_k^- + K_k (z_k - h(\hat{x}_k^-, 0)) \quad (11)$$

$$P_k = (I - K_k H_k) P_k^-, \quad (12)$$

where H_k is the Jacobian of $h(\bullet)$ with respect to the estimate of x at time k :

$$H_k = \begin{bmatrix} \frac{1}{\lambda} & 0 & 0 & 0 & -\frac{x_0}{\lambda^2} \\ 0 & \frac{1}{\lambda} & 0 & 0 & -\frac{y_0}{\lambda^2} \\ 0 & 0 & \frac{1}{\lambda} & 0 & -\frac{x_1}{\lambda^2} \\ 0 & 0 & 0 & \frac{1}{\lambda} & -\frac{y_1}{\lambda^2} \end{bmatrix}, \quad (13)$$

where $\lambda = 1 + z\beta$. Finally, the matrix V is the Jacobian of $h(\bullet)$ with respect to v , and R_k is our measurement noise covariance at time k .

In this formulation, the general assumptions are: w is a Gaussian random vector with $p(w_k) N(0, W Q_k W^T)$, and v is also Gaussian $p(v_k) N(0, V R_k V^T)$. For more detail, see [31, 33].

6 Occlusion Detection and Reasoning

The majority of tracking systems for non-rigid objects handle only isolated objects. Occlusions present a major problem. Researchers have tried to address this problem by using multi-camera tracking, range, stereo, *etc.*

In order to extract robustly trajectories of multiple moving objects, we need to manage the possibility of occlusions. In order to resolve the occlusion and keep track of



Figure 4: Tracking example: input sequence, and blob segmentation results.

occluded objects, we need a good estimator of position and velocity.

Our occlusion detection routine predicts future locations of objects, based on current estimates of 3D velocity and position. We use the system dynamics formulation of Sec. 5.4:

$$x_{k+\delta t} = A(\delta t)_k x_k, \quad (14)$$

and a prediction in the observation is computed:

$$z_{k+\delta t} = h(x_{k+\delta t}). \quad (15)$$

In this way, it is easy to predict a collision of two or more objects in the camera plane. If a collision of the objects is detected, we assume that occlusion is probable. Our confidence in this probable collision is determined by the current estimate of error given by the EKF. We use a simple threshold to tell when the *occlusion reasoning* mechanism is used. In this case the *tracker structures* corresponding to the given objects are tagged for this potential event.

A false alarm can occur, so a recovery mechanism is used in which the system checks for reconfirmation of occlusion every time after occlusion was predicted.

Occlusion is confirmed by using both the estimated bounding box parameters and the binary maps for the objects being tracked. The effect of an occlusion will be the merging of the blobs (binary maps) for the objects. After this occurs, the system uses the EKF prediction to update the object position and velocity. The merged blob provides a good way to constraint the possible location of any of the occluded objects. It is used to detect errors in prediction due to noise or motion that does not meet our assumptions.

While an occlusion is in progress, the system expects that merged blobs will eventually separate. Occlusion time can be estimated using the EKF estimates of 3D velocity and position along with the computed area of the blob. Furthermore, the division of merged blobs is provides strong evidence for the end of an occlusion. Correspondence can be solved using object locations predicted through the EKF.

7 Results

The system was implemented and experiments were conducted on a SGI O2 R5K workstation. For all experiments we used either a consumer hand held video camera, or the standard SGI O2 uncalibrated camera recording at 30 frames/sec (320x240 pixels, color). In order to test the system, we collected 20 sequences of people walking and occluding each other. We show some representative examples. In order to show the robustness to parameter setting, all experiments use exactly the same parameter configuration. The scene learning phase took 1.5 sec in each example (45 frames).

Our first example, in Figs. 4, consists of a 10 second multiple body walking sequence. It shows the standard behavior of the system when motions are roughly on a linear path. Note that trajectories are not parallel to the camera plane. This causes non-linear paths in the image. There are two color-coded boxes and one white box surrounding every tracked object. The box with thicker lines corresponds to the object estimated position. The second color box gives an estimate of the position 1/3 sec. ahead based on the current estimate of the velocity, using a first order model. During the whole sequence, people were tracked and segmented accurately. Occlusion was predicted, detected and handled properly by estimating positions and velocities followed by projecting these estimates to the image plane.

Fig. 5 shows the normalized coordinate frames extracted. The moving object is resampled in a canonical frame throughout the tracking sequence, despite changes in scale and position. The estimated bounding box is used to resample the moving blob to a canonical view that can be used as input to motion recognition modules.

Fig. 6 shows the evolution in the estimates of 3D velocity and position given by our EKF. Body 2 entered the scene about 55 frames after body 1. The estimates visually agree with the motions shown in the sequence. Note that we are estimating $z\beta$ not z . Scale in depth estimates is not necessarily the same as in x,y estimates.

Finally, given the information recovered, it is possible to construct a top-view map, showing the trajectories of the

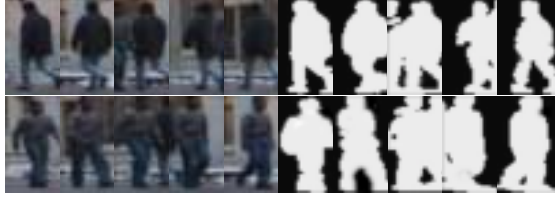


Figure 5: Normalized views of the 2 bodies in the sequence, one row per body. 6 views with their respective regions of support.

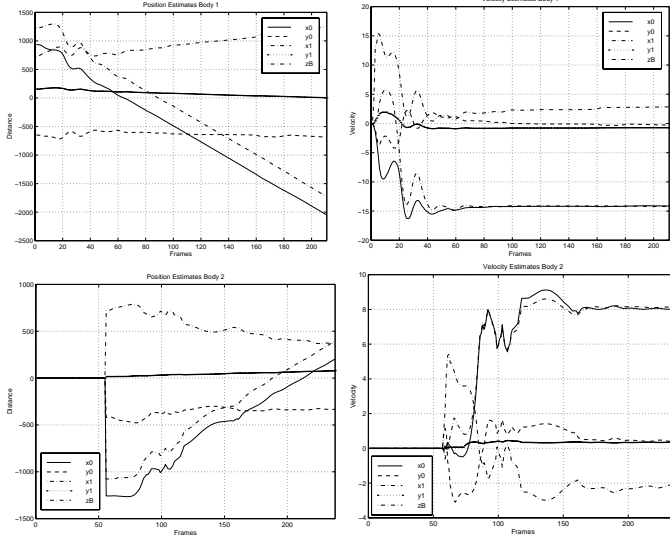


Figure 6: Estimated positions and velocities in 3D given by the EKF. The top row shows position and velocity predictions for body 1. The bottom row shows predictions for body 2. Note that body 2 appears in scene about 55 frames after body 1.

bodies Fig. 7. The first body moves from right to left, while the second body moves in the opposite direction. While the early position estimates are noisy, after 20-40 frames the EKF converges to a stable estimate of the objects' trajectories.

The system has been tested on dozens of examples, with motions varying in depth, direction, and velocity. The system can successfully track and recover positions and velocities despite motion towards the camera, and/or occlusions. In general, the model works well, as long as there is not a sudden change in direction during an occlusion.

Our next example shows a sequence where, during occlusion, one of the objects suddenly changes its direction. If there were no occlusion, the EKF would adapt its estimate to the new direction. But because the change in direction occurs during occlusion, the EKF estimate will be in error and could obviously lose the track, as seen in the estimates shown during and after occlusion in Fig. 8. Speed of re-convergence depends on current error covariance, expected measurement noise and model noise. In this example even though there is a sudden change in direction,

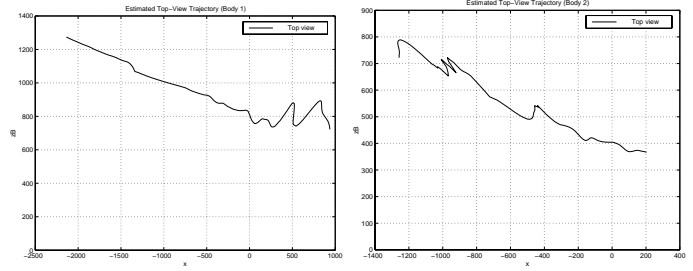


Figure 7: Recovered (top view) motion. Note that motion of body 1 is from right to left. Body 2 goes in the opposite direction. Estimates at the beginning of the sequence are not very accurate, because error covariance in the EKF is higher. Uncertainty is reduced as enough samples are given.

the tracker can in some cases recover by solving for correspondence directly. This is a result of the characteristics of the situation but it is useful to illustrate how in some cases it is possible to recover from inaccuracies in the estimates during occlusion.

8 Discussion

We now discuss and provide some insights of the improvements made by our approach, its strengths and its weakness. The system can accurately track multiple subjects in most situations and with no user interaction required, mainly due to the power of the detection mechanism and the motion estimation techniques. Handling occlusions using motion trajectories is inherently hard if the expected motion model is violated during critical estimation stages as was shown in Fig. 8. The use of other features would be a natural extension to provide a more robust mechanism.

Recovery of trajectories of non-rigid objects has been normally approached assuming no structure (*e.g.* tracking subjects seen as points) or using domain constraints (*e.g.* ground plane known or collapsing one dimension by using a top-view of the scene). For motion analysis, enough information about the shape of the objects and how they evolve is needed. Many previous approaches cannot provide this, even though they make the recovery of trajectories a lot simpler.

The system's time complexity depends mainly in the number of pixels in the input image and the number objects being tracked. Most of the CPU time is consumed calculating the distance between incoming images and the learned scene, comparing it with previous estimates of the region of support, performing the segmentation, applying the filters to the images, and recursively solving for the estimates in the EKF for every object. The experimental system averaged 5 fps.

For convergence, the EKF needed about 40 frames on average in our experiments. The performance and generality of the system can be improved by using an Interacting Multiple Model approach (IMM) [8, 7]. In this approach, n EKF's with different dynamics properties are run together



Figure 8: Tracking results, non-linear trajectories. Bodies are tracked, but prediction is less accurate because of sudden change in direction during occlusion.

and the system determines which one better describes the observations. This could allow for the estimation of positions when we want to consider different model dynamics.

Acknowledgments

This work is sponsored in part through grants from the National Science Foundation: Faculty Early Career Award #IIS-9624168, and CISE Research Infrastructure Awards #CDA-9623865 and #CDA-9529403. Rómer Rosales is sponsored in major part by a doctoral scholarship from the Venezuelan Government CONICIT-Universidad Centro Occidental Lisandro Alvarado program.

References

- [1] A. Azarbayejani and A. Pentland. Recursive estimation of motion, structure, and focal length. *PAMI*, 17(6), 1995.
- [2] A. Azarbayejani and A. Pentland. Real-time 3d tracking of the human body. In *Image Com*, 1996.
- [3] A. Baumberg and D. Hogg. Learning flexible models from image sequences. In *ECCV*, 1994.
- [4] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik. A real-time computer vision system for measuring traffic parameters. In *CVPR*, 1997.
- [5] M. Bichsel. Segmenting simply connected moving objects in a static scene. *PAMI*, 16 (11):1138-1142, 1994.
- [6] A. Blake, R. Curwen, , and A. Zisserman. A framework for spatiotemporal control in the tracking of visual contours. *IJCV*, 11(2):127-145, 1991.
- [7] H. Blom. An efficient filter for abruptly changing systems. In *Proc. Conf. on Decision Control*, 1984.
- [8] K. Bradshaw, I. Reid, and D. Murray. The active recovery of 3d motion trajectories and their use in prediction. *PAMI*, 19(3), 1997.
- [9] C. Bregler. Learning and recognizing human dynamics in video sequences. In *CVPR97*, 1997.
- [10] T. Darrell and A. Pentland. Classifying hand gestures with a view-based distributed representation. In *NIPS*, 1994.
- [11] J. Davis and A. F. Bobick. The representation and recognition of human movement using temporal templates. In *CVPR*, 1997.
- [12] N. Ferrier, S. Rowe, and A. Blake. Real-time traffic monitoring. Technical report, Robotics Research Group, Oxford U., 1996.
- [13] D. Gavrilu and L. Davis. Tracking of humans in action: a 3-d model-based approach. In *Proc. ARPA Image Understanding Workshop, Palm Springs*, 1996.
- [14] D. Hogg. *Interpreting Images of a Known Moving Object*. PhD thesis, University of Sussex, 1984.
- [15] S. Intille and A. F. Bobick. Real time close world tracking. In *CVPR*, 1997.
- [16] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *ECCV*, 1996.
- [17] G. Johansson. Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics*, 14(2): 210-211, 1973.
- [18] S. Ju, M. Black, and Y. Yacoob. Cardboard people: A parameterized model of articulated image motion. In *Proc. Gesture Recognition*, 1996.
- [19] I. Kakadiaris, D. Metaxas, and R. Bajcsy. Active part-decomposition, shape and motion estimation of articulated objects: A physics-based approach. In *CVPR*, 1994.
- [20] D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. Technical report, U. C. Berkeley, 1994.
- [21] M.W. Krueger. *Virtual Reality II*. Addison-Wesley, 1990.
- [22] L. Matthies, T. Kanade, and R. Szeliski. Kalman filter based algorithms for estimating depth from image sequences. *IJCV*, 3(3):209-236, 1989.
- [23] S. Niyogi and E. H. Adelson. Analyzing and recognizing walking figures in xyt. In *CVPR*, 1994.
- [24] A. Pentland and B. Horowitz. Recovery of non-rigid motion and structure. *PAMI*, 13(7):730-742, 1991.
- [25] R. Polana and R. Nelson. Low level recognition of human motion. In *Proc. IEEE Workshop on Nonrigid and Articulate Motion*, 1994.
- [26] B. Rao, H. Durrant-Whyte, , and J. Sheen. A fully decentralized multi-sensor system for tracking and surveillance. *Int. J. of Robotics Research*, 12(1), 1993.
- [27] J. M. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. In *ICCV*, 1995.
- [28] D. Reynard, A. Wildenberg, A. Blake, and J. Marchant. Learning dynamics of complex motions from image sequences. In *ECCV*, 1996.
- [29] K. Rohr. Towards model-based recognition of human movements in image sequences. *CVGIP:IU*, 59(1):94-115, 1994.
- [30] M. Shah and R. Jain. *Motion-Based Recognition*. Kluwer Academic, 1997.
- [31] H.W. Sorenson. Least-squares estimation: From gauss to kalman. *IEEE Spectrum*, Vol. 7, pp. 63-68, 1970.
- [32] R. Szeliski and S. Bing Kang. Recovering 3d shape and motion from image streams using non-linear least squares. In *CVPR*, 1993.
- [33] G. Welch and G. Bishop. An introduction to the kalman filter., Technical Report TR 95-041, Computer Science, UNC Chapel Hill, 1995.
- [34] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfnder: Real time tracking of the human body. Technical Report TR 353, MIT Media Lab, 1996.
- [35] J. Yamato, J. Ohya, and K. Isii. Recognizing human action in time sequential images using hidden markov models. In *CVPR*, 1993.