IEEE International Conference on Computer Vision, Corfu, Greece, September 1999

Tracking Self-Occluding Articulated Objects in Dense Disparity Maps

Nebojsa Jojic*, Matthew Turk** and Thomas S. Huang*

*ECE Dept. and Beckman Institute
University of Illinois at Urbana-Champaign
jojic@uiuc.edu

**Microsoft Research
Redmond, WA
mturk@microsoft.com

Abstract

In this paper, we present an algorithm for real-time tracking of articulated structures in dense disparity maps derived from stereo image sequences. A statistical image formation model that accounts for occlusions plays the central role in our tracking approach. This graphical model (a Bayesian network) assumes that the range image of each part of the structure is formed by drawing the depth candidates from a 3-D Gaussian distribution. The advantage over the classical mixture of Gaussians is that our model takes into account occlusions by picking the minimum depth (which could be regarded as a probabilistic version of z-buffering). The model also enforces articulation constraints among the parts of the structure. The tracking problem is formulated as an inference problem in the image formation model. This model can be extended and used for other tasks in addition to the one described in the paper and can also be used for estimating probability distribution functions instead of the ML estimates of the tracked parameters. For the purposes of real-time tracking, we used certain approximations in the inference process, which resulted in a real-time two-stage inference algorithm. We were able to successfully track upper human body motion in real time and in the presence of self-occlusions.

1. Introduction

Tracking non-rigid motion in image sequences has been of great interest to the computer vision community. The problem is generally divided into studies of two categories of motion: deformable object motion and the motion of an articulated object. The latter is of great interest to the HCI (human computer interaction) community as the human body is an articulated object. The current commercially available motion capture systems are either based on magnetic or optical trackers that require the subject to wear a special suit with markers on it, or even to be attached to the system by cables. A more general solution based on passive sensing would be more convenient, less constraining, and attractive for a variety of uses.

There have been several approaches to human body tracking, ranging from detailed model-based approaches [1,2] to the simplified, but faster statistical algorithms [3] and cardboard models [4]. In [3], 2-D tracking based on Gaussian blobs has been proposed. To track human motion in full 3-D, an extension of this approach based on the input from two cameras was proposed in [5] and tested on upper-body tracking. However, only the hands and head are tracked while the position and orientation of the torso and lower and upper arms is ambiguous. The two cameras are not used to calculate a dense disparity map, but rather to estimate 2-D blob parameters in each image. In [6] an Extended Kalman Filter imposes articulation constraints on the rest of the body which provides a guess about the full posture, but measuring just three points on a human body does not provide enough information for unambiguous posture tracking. Knowledge of the dynamics of human motion is believed to be helpful for tracking [6,7].

In [15], a nice model-based approach to tracking self-occluding articulated structures is proposed. However, the algorithm is based on template matching and is sensitive to lighting changes. One of the great advantages to stereo cues is that the disparity computed based on correlation is less sensitive to the intensity changes.

There have been reports on several simple but fast tracking schemes based on stereo [8] or an integration of stereo with other cues [14]. These papers dealt primarily with head tracking and did not model the articulation constraints and self-occlusions.

In this paper, we propose a tracking algorithm that uses the input from two cameras approximately 8cm apart (Section 2). The disparity map is computed at frame rate by commercially available software [8]. The tracking algorithm is based on a *generative statistical model* of image formation, specifically tuned to rough but fast tracking in the presence of self-occlusions among the articulated 3-D Gaussian models (Sections 3 and 4). This model falls into a broad category of graphical models (Bayesian networks) that have been very successful in diverse applications at formalizing generative processes in ways that allow probabilistic inference [9]. In our case, a maximum likelihood estimate of the posture of an

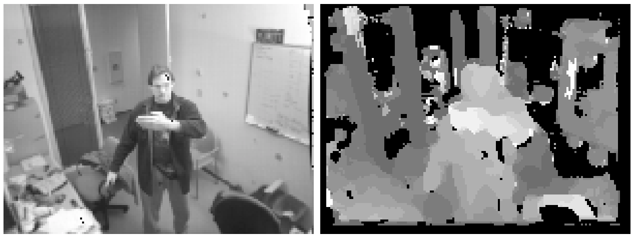

Figure 1: Left intensity image and the disparity map. Black indicates no good data.

articulated structure is achieved by a simplified, but very fast inference algorithm that consists of two stages (Section 5). In the first stage, the disparity map is segmented into different parts of the articulated structure based on the estimated state of the Gaussian mixture using the maximum likelihood principle with an additional mechanism for filling in the missing data due to occlusions. Then, the statistical properties of the individual parts are re-estimated. In the second stage of the inference algorithm, an Extended Kalman Filter (EKF) enforces the articulation constraints and can also improve the tracking performance by modeling the dynamics of the tracked object (as in [6,7]). In Section 6, we describe our experiments on tracking upper human body motion and give our conclusions.

2. Real-time Disparity Map Computation

In our experiments, we used the commercially available real-time stereo system built by SRI [17]. Currently, this system runs at rates of up to 90Hz with image resolution of 160x120, and gives for each pixel the disparity between the images from a stereo pair. The following equations describe the relationship between the 3D coordinates [*x y z*]' of a point imaged by the stereo pair, the coordinates of the point's perspective projection onto the image plane [*X Y*]' and the disparity in the two images D(*X,Y*):

$$\begin{aligned} &x = Xz/f, \quad y = Yz/f, \qquad z = bf/D(X,Y), \\ &X = xf/z, \quad Y = yf/z, \quad D(X,Y) = bf/z \end{aligned} \qquad (1)$$

b denotes the baseline length, and *f* is the focal length of the cameras. In Fig. 1, an example of a disparity map D(X,Y) is given.

3. Tracking Articulated Motion

An articulated object's posture can be parameterized in many different ways. One possibility is to use a redundant set of parameters such as the position and orientation parameters of individual parts and impose the constraints in a Lagrangian form [1,2,6]. Another approach is to use the kinematic chain equations and select parameters that are not redundant, such as the orientations of the individual parts and the position of a reference point [10,11]. We used the latter type of parameterization.

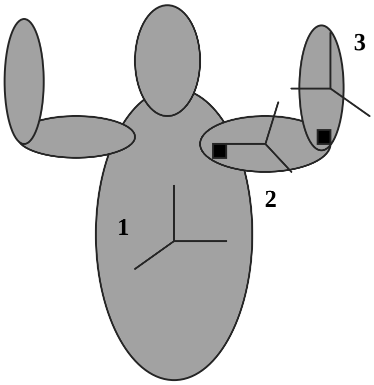

Figure 2: Articulated model – the local coordinate systems and joints.

The 3-D posture of an articulated structure is defined by

$$\alpha = \{\mu_1, Q_1, Q_2, ..., Q_K\} \qquad (2)$$

where μ_1 is the center of the reference part, and Q_k is the quaternion representation of the global orientation of the *k*-th part. In particular, a position vector in the local coordinate system is transformed into the position vector in the global coordinate system by

$$p_{global} = R(Q_k)\, p_{local} + \mu_k \,,$$

where μ_k is the position of part *k* (and the origin of the coordinate system attached to that part) and $R_k = R(Q_k)$ is the rotation matrix corresponding to the quaternion Q_k [12].

Given the posture parameters α, the individual positions and orientations of the parts of the articulated structure can be computed. If $J_i^{(k)}$ is the position of the *i*-th joint in the **local** coordinate system of the *k*-th part, we can write the kinematic chain equations like the one illustrated in Fig. 2 as:

$$\mu_3 = \mu_1 + R_1 J^{(1)}_{shoulder} - R_2 J^{(2)}_{shoulder} + R_2 J^{(2)}_{elbow} - R_3 J^{(3)}_{elbow} \qquad (3)$$

The positions of the joints in their local coordinate systems do not change over time.

To predict the range data, in addition to the articulation model, we need the models of individual parts, which could range from complex geometrical models [1,2] to statistical models such as the one in [3]. Shortly, we shall define a particular statistical body part model and the model of the image formation process, but for the moment, let us go on with the definition of the tracking problem and its general solution.

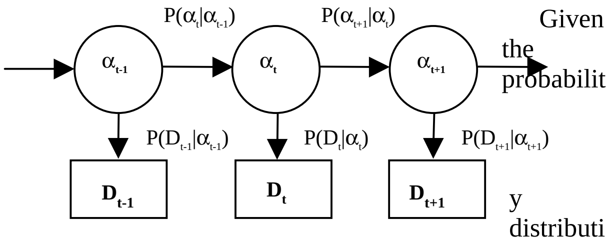

Figure 3: Markov chain representation of the tracking problem.

on of the parameters α conditioned on the past of the system $P(\alpha_t|\ past)$, and the probabilistic image formation model $P(D_t|\alpha_t)$, where D_t is the observed data at time t, the dynamics of the system can be represented by a Markov chain as shown in Fig. 3. We assume that the state of the system α_t can be augmented (by derivatives, for example), so that the previous state α_{t-1} contains all the necessary information for conditioning the distribution of α_t, i.e., $P(\alpha_t|\ past)= P(\alpha_t|\ \alpha_{t-1})$.

The tracking problem consists of finding the sequence $\{\alpha_t\}$ that maximizes the likelihood of the observed data D_t, which in our case is the disparity map adhering to the imaging equations in Eq. 1. In the next two sections, we define a statistical image formation model $P(D_t|\alpha_t)$ (Fig. 4), and an approximate, but fast inference algorithm that updates the parameters of the articulated model based on the current disparity map.

4. Image Formation Model

In the previous section, we left out the model of the individual parts of the tracked articulated structure. Since we are motivated by real-time applications, we are willing to sacrifice the precision of the model to reduce the computational complexity. Simple blob-type statistical models of regions in an image have been used in similar situations in the past. Our blobs are defined in the 3-D space on the points satisfying the imaging equations given in Eq. 1. Each pixel in the image is associated with an observation O that consists of a color c=[Y U V]' and a 3-D position p=[x y z]'. Of course, other features could be added. We make an assumption that the observation probability density is normal (Gaussian):

$$P_k(O) = N(O;\mu_k,K_k) = \frac{e^{-\frac{1}{2}(O-\mu_k)^T K_k^{-1}(O-\mu_k)}}{(2\pi)^{m/2}\left|K_k\right|^{1/2}} \qquad (4)$$

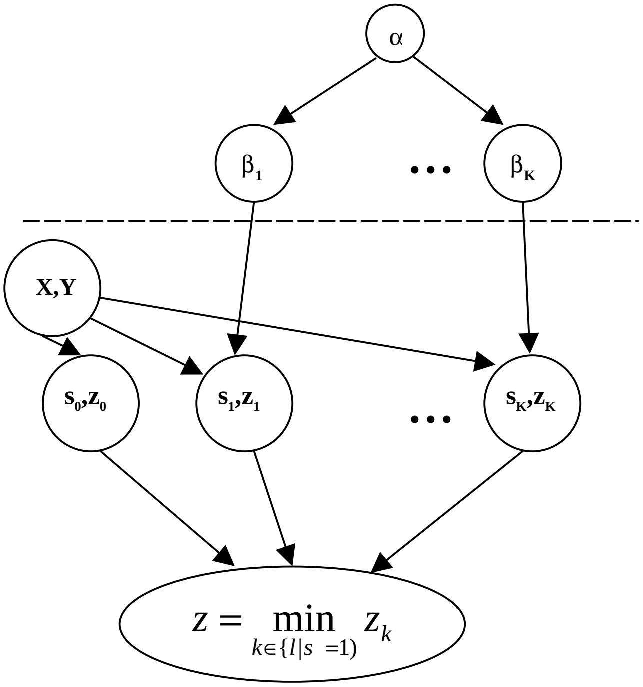

Figure 4: Bayesian net representation image formation

The color and position parts of the observation vector are not in general correlated (or rather, this correlation can not be adequately captured by a simple Gaussian model), so the covariance matrix can be assumed to be block-diagonal. In this case the probability distribution can be expressed as the product of two Gaussians. In the following discussion, as well as in our experiments, the color information is not used. However the separability of the probability distributions would make it easy to add color information as described in [3]. We concentrate instead on using the range data which is directly obtained from the disparity map: $z(X,Y)=bf/\mathrm{D}(X,Y)$, and we shall assume that:

$$O(X,Y) = p = [x\ y\ z]'.$$

We note that the points available from the disparity map are on the frontal surface of the imaged body (Fig. 1). Therefore, our Gaussian blobs are meant to model the points on the parts of the body facing the camera. The parameters of the Gaussian distribution $\beta_k=(\mu_k,\ K_k)$ capture the position, size and orientation of body part k. The position is determined by the mean μ_k, while the orientation and size are captured in the covariance matrix K_k that can be decomposed as:

$$K_k = R(Q_k)'\Lambda_k R(Q_k) \qquad (5)$$

Λ_k is a diagonal matrix whose non-zero elements are the eigenvalues of the covariance matrix that determine the size of the body part model along the three orthogonal directions aligned with the local coordinate system attached to the part. $R(Q_k)$ is the rotation matrix that was defined in Section 3. Since we are modeling the data directly in 3-D, it can be assumed that Λ_k (blob size) remains constant in time. This is a significant advantage over the 2-D blob trackers that have to allow the change of both the eigenvalues and eigenvectors of the covariance matrix. By keeping the size parameters fixed, the algorithm becomes more stable.

It can be readily seen that the individual blob parameters $\beta=\{\beta_k\}_{k=1,N}$ are related to the parameters of the articulation model α:

$$\beta = h(\alpha), \quad or \quad \beta_k = h_k(\alpha), \quad k = 1...N \tag{6}$$

where h is the collection of the kinematic chain equations such as Eq. 3, and the collection of decompositions of covariance matrices (Eq. 5). These relationships are nonlinear due to the quaternion parameterization of the rotation matrices, and this nonlinearity cannot be avoided by other parameterizations.

Another way of looking at this probabilistic model is to study the measured value of depth $z=bf/D$ for the pixel (X,Y). As can be seen in Eq. 1, the 3-D position vector of the image point is related to the depth z as:

$$p = zr, \quad where \quad r = \begin{bmatrix} X/f \\ Y/f \\ 1 \end{bmatrix}. \tag{7}$$

Given the blob k and its parameters β_k, the distribution of the depth z along the ray r is given by:

$$P_k(z \mid X,Y,\beta_k) = \gamma \; N(zr(X,Y);\mu_k,K_k) = N(z;\mu_{zk},\sigma^2_{zk}), \tag{8}$$

where γ is a scaling constant. In this equation, index $k=0$ is reserved for the background model, which can either be formed by observing the empty scene prior to the entrance of the subject, or simply by defining a bimodal distribution in which the likelihood of the background class is very large for the points farther than some threshold. In other words, effective background subtraction in range images can be done by simple thresholding, which is what we did in our experiments. One of the big advantages of the depth stereo map as a visual cue is the easier background subtraction (for example, see [8]).

The parameters $(\mu_{zk}, \sigma^2_{zk})$ of the normal distribution of z along the line of sight (r) for pixel (X,Y) can be easily identified in the exponent of the Gaussian $N(zr;\, \mu_k,\, K_k)$:

$$\mu_{zk} = \frac{r' K_k^{-1} \mu_k}{r' K_k^{-1} r}, \quad \sigma_{zk}^{-2} = r' K_k^{-1} r \tag{9}$$

Studying the probability distribution of the depth along the line of sight (r) will soon prove to be useful for elegantly modeling occlusions in a statistical framework. However, normalizing this distribution destroys the sensitivity of the probabilistic model to the distance between the line of the sight and the mean of the blob. In other words, even the blobs whose means are far away from the ray r can have substantial likelihood near the projection of their mean onto this ray. This is a natural result of the dimensionality reduction in Eq. 8, but the lost information can be easily restored by introduction of the class indicator variables $s_k \in \{0,1\}$ for each pixel (X,Y). s_k is a random variable indicating if the appropriate blob takes part in the imaging process. The mixing proportions p_k can be approximated using the definition of the Gaussian blobs in Eq. 4. Value μ_{zk} is the most likely value for the blob k along the line of sight $r(X,Y)$. Thus, the probability that blob k will produce a point at (X,Y) can be defined as:

$$p_k = E(s_k) = \frac{N(\mu_{zk} r;\mu_k,K_k)}{N(\mu_k;\mu_k,K_k)} = e^{-d_k(\mu_{zk} r,\mu_k)}, \tag{10}$$

where $d_k(\bullet,\bullet)$ is the Mahalonobis distance based on the covariance matrix K_k. In terms of the graphical model in Fig. 4, Eq. 10 determines conditional probability $p(s_k|X,Y,\beta_k)$.

The simple mixture of 3-D Gaussians is now substituted by a more complex probabilistic model consisting of the depth distribution along the line of sight for each blob and the a priori probability of each blob for a given pixel. However, we are now able to define the occlusion in our generative model as follows.

In the model of the image formation process, s_k is set to one with the probability p_k. Let $V=\{k \mid s_k=1\}$ be the set of selected classes $(s_k=1)$. For each $k \in V$, a value z_k is drawn from the distribution for the k-th blob (with parameters in Eq. 9). Finally, the range image is formed by picking the 3D blob point **closest** to the image plane:

$$z = \min_{k \in \{l|s_l=1)} z_k, \text{ where} \begin{Bmatrix} z_k \sim P_k(z|X,Y,\beta_k) \\ s_k \sim p_k \end{Bmatrix} \tag{11}$$

We keep $s_0=1$, so that the background model always participates in the imaging process. Eq. 11 describes the deterministic decision in the last stage of Fig. 4.

Given the blob parameters β and the set of selected classes V, the distribution of z for the imaging process of Eq. 11 is given by:

$$P(z \mid X,Y,\beta,V) = \sum_{k \in V} P_k(z \mid X,Y,\beta_k) \prod_{l \in V \setminus \{k\}} G_l(z \mid X,Y,\beta_l) \tag{12}$$

$$G_l(z \mid X,Y,\beta_l) = \int_z^\infty P_l(w \mid X,Y,\beta_k)dw$$

The probability of the set V is given by:

$$p(V \mid X,Y,\beta) = \prod_{k \in V} p_k \prod_{l \notin V} (1 - p_l) \tag{13}$$

The complete image formation process for the pixel (X,Y) can be depicted by the graphical model in Fig. 4. The imaging model is conditioned on the probability density of the postures α given the past of the tracker. The parameters β_k of individual blobs depend deterministically on the parameters α (Eq. 6). The blob parameters affect the stochastic selection of several depth candidates and finally, the minimum of these candidates (the one closest to the imaging system) is selected.

This process has two nonlinearities given by Eqs. 6 and 11. By making appropriate linearizations, it might be possible to derive an Extended Kalman Filter that would use the depth values for each pixel as the measurements to update the state of the articulated model. However, the large number of pixels in the disparity map makes this approach computationally expensive.

In the next section we derive a fast, two-stage approximate solution to the problem of inference in the graphical model of Fig. 4.

5. Tracking Algorithm

In order to achieve real-time speeds and utilize all the available pixels in the disparity image, we propose a two stage process in updating the state of the articulation model α. These two stages deal with the two halves of the graphical model shown separated by a dashed line in Fig. 4.

5.1 Re-estimation of the blob parameters β

Without the nonlinear process of Eq. 11, the lower half of Fig. 4 would represent an ordinary mixture of Gaussians (with distributions given by Eq. 4), which could be trained using several iterations of the EM algorithm [16]. In the exact EM algorithm, the pixels in the disparity map are assigned a probability for each blob *k* in the E-step, thus softly clustering the data points. Alternatively, hard clustering could be performed, where each pixel is assigned to a single class. This speeds up the subsequent computation as each pixel is included in the statistics of a single class only. Sacrificing the optimal E-step is less serious when plenty of data is available, which is the case in our experiments, as the subjects occupy a significant portion of the image. A support map defining the hard clustering of the data is computed as:

$$S(X,Y) = \arg\max_k \log(P_k(x,y,z)) \,, \qquad (14)$$

where the likelihood is computed using the current estimate of the blob parameters β.

Given the support map, the blob parameters can be re-estimated using a well-known formula for Gaussian training (M step):

$$\hat{\mu}_k \approx \frac{1}{N_k} \sum_{S(X,Y)=k} O(X,Y) = \frac{1}{N_k} \sum_{S(X,Y)=k} zr \,,$$

$$\hat{K}_k \approx \frac{1}{N_k} \sum_{S(X,Y)=k} (O-\hat{\mu}_k)(O-\hat{\mu}_k)' , \qquad (15)$$

In the past, similar techniques have been used for 2-D tracking. In [3], for each new frame, only a single iteration of Eq. 14 and 15 is applied, starting from the predicted values of the blobs' parameters. The prediction is based on Kalman filtering.

5.2 Occlusion detection and filling the missing data

Simply iterating Eqs. 14 and 15 without taking into account Eq. 11 creates numerous occlusion problems. The main problem stems from the fact that the blobs, though 3-D objects, compete on the 2-D image plane. As result of the application of Eqs. 14 and 15, the occluding blob will push the occluded blob away, considerably affecting its mean and covariance matrices. The appropriate solution is based on maximizing the probability in Eq. 12, averaged over all possible sets V and blob parameters β. It can be shown that with certain approximations, this solution reduces to inserting into the Eqs. 14 and 15 the estimate of the data missing due to occlusions. For the sake of brevity we choose to omit this derivation, since the intuitive explanation that follows better captures the essence of our occlusion handling approach.

Eqs. 14 and 15 are an approximation based on hard clustering to the exact EM algorithm and they take into account the visible data when updating the blob's parameters. To handle occlusions, we continue along the same lines of hard decision approximations. After the pixel (*X,Y*) has been assigned to a class $k^*=S(X,Y)$, we can study the image formation model of the previous section to estimate which blobs had made it into the selected set $V=\{k|s_k=1\}$ before the minimum value $z(X,Y)=z_{k^*}$ was picked. Instead of the soft decision based on the mixing probabilities, we make a hard decision by thresholding the mixing probabilities in Eq. 10 and estimate the set V as

$$\hat{V} = \{k \mid p_k > p_T, \mu_{zk} r > z\} \,, \qquad (16)$$

where p_T is a threshold on the mixing probabilities, and p_k and μ_{zk} in Eqs. 9 and 10 are computed using the current estimates of the blob parameters β_k. In essence, the blobs in the estimated set *V* would have been likely to produce a point at the location X,Y (since p_k is high), but were occluded by the winner *k** (as the measured depth is smaller than the most likely depths the blobs in V would have produced). Not knowing which value z_k was drawn from *k*-th distribution (since the point was occluded), our best guess is the mean of the distribution of z along the line of sight *r*, i.e., $\hat{z}_k = \mu_{zk}$, where μ_{zk} is given in Eq. 9. In other words, the missing 3D point is assumed to be $\mu_{zk}r$. Now, the estimation equations can be rewritten as:

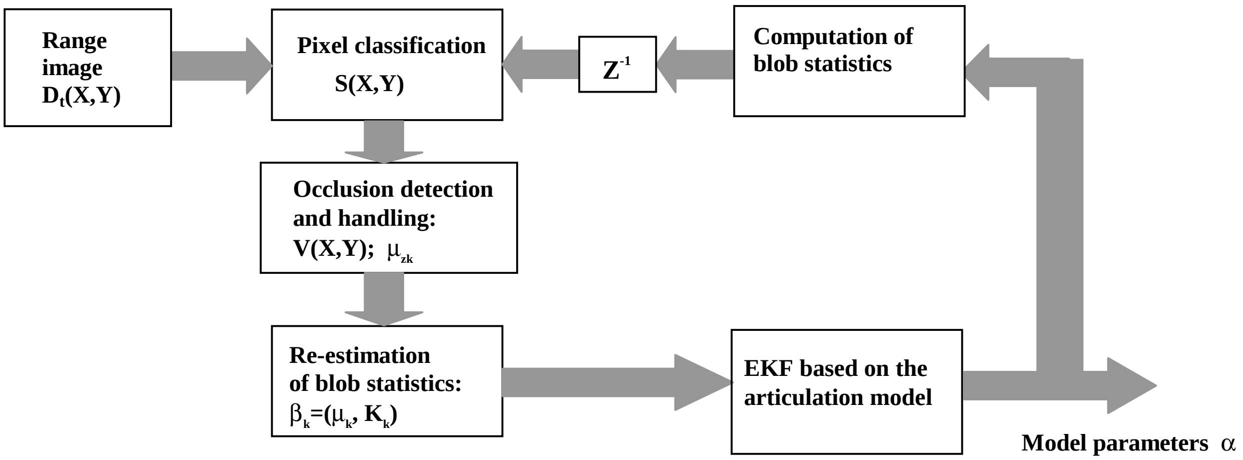
Range image $D_t(X,Y)$
Pixel classification S(X,Y)
Z^{-1}
Computation of blob statistics
Occlusion detection and handling: V(X,Y); μ_{zk}
Re-estimation of blob statistics: $\beta_k=(\mu_k, K_k)$
EKF based on the articulation model
Model parameters α

Figure 5: The block diagram of the tracking algorithm.

$$\hat{\mu}_k \approx \frac{1}{N_k}\left(\sum_{S(X,Y)=k} zr + \sum_{k\in V(X,Y)-\{S(X,Y)\}} \mu_{zk} r \right)$$
$$\hat{K}_k \approx \frac{1}{N_k}\left(\begin{array}{l} \sum\limits_{S(X,Y)=k} (zr-\mu_k)(zr-\mu_k)' + \\ + \sum\limits_{V(X,Y)-\{S(X,Y)\}\ni k} (\mu_{zk} r-\mu_k)(\mu_{zk} r-\mu_k)' \end{array} \right), \quad (17)$$

where N_k is the total number of pixels used for updating the parameters of the blob *k*. In these equations, the observed values are included in the same fashion as in the regular M step of the EM training of a Gaussian mixture. However, these equations are also supplemented by the data that might have been lost due to occlusions. Starting with the predicted blob parameters $\beta_k(t|t\text{-}1)$ and applying these equations to the new frame, we get new estimates of these parameters that take into account all pixels in the disparity map in an approximate, but computationally very efficient manner. These new estimates are forwarded to the second stage of the estimation algorithm (the upper half of Fig. 4) whose goal is to re-estimate the posture α for the current frame and predict the posture in the next time step, thus providing the estimate of individual blob parameters for the next frame.

5.2.1 On the exact EM algorithm for inference of β

From the above discussion and Fig. 4 it is not difficult to see that a better inference in the lower part of the formation model is based on iterating the soft-clustered versions of Eqs. 14, 16 and 17. In particular, in the first part of each of the two estimate equations in Eq. 17, the measured depth *zr* (see Eq. 7) should be scaled by $p(k|x,y,z)=P_k(x,y,z)/\Sigma P_i(x,y,z)$, where P_k is given in Eq. 4. In the second part (occluded data filling), the estimated depth should be scaled by the probability p_k of Eq. 10 and thus the set *V* should include all blobs.

Such an approach would lead to a better estimate of the blob parameters in the first stage of our inference algorithm, but even though only three or four iterations of EM would be sufficient, this would hurt the real-time performance of the system. Just using soft clustering instead of the hard decisions in Eqs. 14 and 16 and still performing only a single re-estimation of blob parameters also reduces the speed a bit (as more multiplications become necessary), but this overhead is much less significant. In our experiments, to achieve the best frame rate, we used Eqs. 14 and 16 as they are. Furthermore, only a single iteration of Eqs. 14 and 17 is applied in each frame, as we rely on the good estimate of the blob parameters provided by the EKF described in the next section.

5.3 The Extended Kalman Filter (EKF) with Implicit Articulation Constraints

Since we are tracking dynamical systems, the transition of the system (horizontal direction in Fig. 3) can be captured by a linear system (for larger orders, we need to augment the state α with its derivatives). We assume that Eq.17 represents noisy measurements of the true values of β. Therefore, the estimation of articulation model parameters can be done in the Extended Kalman Filtering framework [13], where the transition is captured by a linear equation (with added noise) and the measurement equations are nonlinear and noisy:

$$\begin{aligned} \alpha_t &= F\alpha_{t-1} + noise \\ \hat{\beta}_t &= h(\alpha_t) + noise \end{aligned} \quad (18)$$

where *h* is given by Eqs. 3, 5, and 6 in Section 2. The EKF linearizes *h* around the current estimate of α to create the Jacobian matrix that plays the role of the measurement matrix in the regular linear Kalman Filter.

Our parameterization of the articulated model is non-redundant, so that each combination of the parameters corresponds to a posture in which the blobs are connected at the joints. Thus, the EKF enforces the articulation constraints on the blob parameters. Matrix F captures the dynamics of the system, but in the simplest version it can be set to the identity matrix.

The simple block diagram in Fig. 5 summarizes the tracking algorithm described in this section.

6. Experimental Results and Conclusion

In order to perform experiments on our system, we used a simple initialization algorithm in which the subject is expected to assume a certain pose based on marks in the image, after which the different regions in the image are assigned to different body parts – e.g., the head, torso, upper arm, and forearm. The statistical parameters of the initial blobs are computed, and the eigenvalues of the covariance matrices are found. The joints are also assumed to be at certain fixed positions in the image plane during initialization. This initialization routine proved to be sufficient for testing the overall algorithm, but the effects of bad initialization are often visible. However, the model initialization problem can be solved with some of the heuristic techniques used by other tracking algorithms found in the literature.

In the first series of experiments, we used a simple two-part model of the upper human body consisting of the head and the torso blobs (Fig. 6). The crude initialization technique worked sufficiently well for this purpose. The tracker is insensitive to the scale change and even recovers from complete breakdowns such as the ones that inevitably occur when the subject wanders completely out of the field of view and then returns. To make the recovery from breakdowns faster, gravity and antigravity forces can be applied. The gravity force, pulling the blob down, is applied to the torso blob, while the antigravity force, pulling the blob up, is applied to the head blob. Apart from robustness to scale change, the advantage of this type of tracking is its insensitivity to illumination changes and the changes in the far background. The tracker is not dependent on the color information, though it can be easily incorporated in the model as an additional modality. Furthermore, in addition to segmenting the head and torso in the images, the tracker also gives their 3D positions. Using the two connected blobs in this fashion proved to considerably add to the robustness of the tracker. A single blob tracker could wonder off the head and down to the torso. The EKF measurement equations assume constant eigenvalues of the covariance matrix, thus preventing the blobs from "eating each other." The tracker operates at around 20Hz (including disparity map computation) on a 333MHz Pentium, and this speed was achieved without much effort invested in optimization. This tracker can be combined with existing (less robust but more precise) face trackers and face pose estimators to create a robust and precise system.

In the second set of experiments (Fig. 7), we tracked an articulated structure consisting of 4 blobs representing the head, torso, lower arm and the upper arm. In this case, the sensitivity to the initial model parameters becomes an issue. However, once the initial model is properly selected (by experimenting, taking body measurements or combining the above with the joint refinement), the tracking becomes reliable and the tracker handles self-occlusions well – such as when the forearm occludes part of the upper arm or torso. In Fig. 7 we demonstrate tracking that uses a non-perfect model which resulted from initialization based on a single frame. It can be seen that the model is not of the right size, which creates some artifacts, but the tracker remains on the body at all times (see also the movies on the web page).

There is still more effort to be invested in proper selection of the measurement noise model and the state transition equations in the EKF, which we expect to further improve the robustness and speed of tracking. The current speed is limited by the rather conservative estimates of the measurement noise. The computation (in our non-optimized code) limits the four-blob tracker's speed to around 10-15Hz, but the assumption of high measurement noise prevents tracking of rapid motions; the user must still move slowly.

Overall, the system proved to be insensitive to depth-dependent scale changes, as this is taken into account in Eq. 1. In fact, scale changes in the image even helps the tracker indirectly, as the size parameters of the 3-D blob model are fixed, and the scale change due to the perspective projection becomes an additional depth cue. The system operates well under serious self-occlusions as in Fig. 7. Since it is based only on range data and a crude statistical model, the system has difficulties estimating a rotation about an axis parallel to the image plane, even though it segments the disparity image correctly. In the current framework, the blobs represent the frontal surface points that were imaged by the cameras, and therefore always float on the frontal parts of the body surface, like some sort of an articulated mask. This is sufficient for estimating the body posture, as many applications treat the arms as sticks, i.e., only the direction of the major eigenvector is important (e.g., for pointing), while the torso's orientation is highly constrained by the shoulder and neck joints. These constraints help in correctly estimating all rotational degrees of freedom. However, the current model can not capture the head pose sufficiently well (namely, the degree of rotation around the vertical axis), but it can segment the head for further processing by other algorithms. We plan to add a more complex statistical model that captures the visible, frontal part as well as the back part of the head. Better pose estimate will be possible with such a model and the additional constraints from the optical flow in the intensity images (which captures well the image motion due to the object rotation around an axis parallel to the image plane; see the approach in [10]).

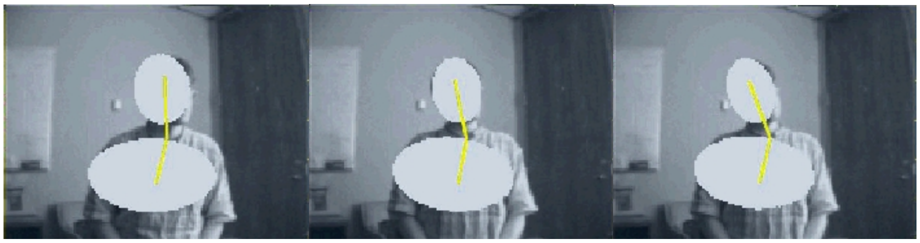

Figure 6: Three frames in tracking of the connected head and torso blobs.

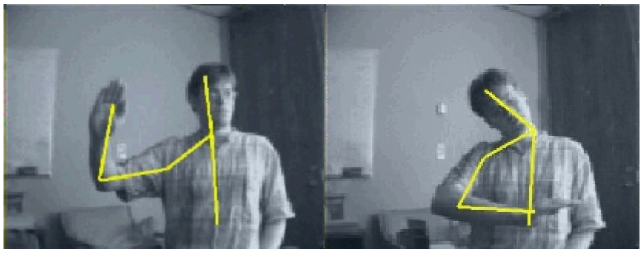

Figure 7: Two frames demonstrating the initialization and tracking in the presence of self-occlusions. One set of lines connect the neck, shoulder and elbow joints, while the line segments on the lower arm, head and torso extend from the joints in the direction of the appropriate blob centers but go further so that their lengths represent the sizes of the blobs.

The potential applications of an articulated tracker such as this one are in vision-based interfaces, such as tracking people, detecting pointing gestures, and computing the direction of pointing. The captured motion can also be used to animate computer graphics characters or the avatars in video conferencing software and VR applications. The tracker can also be used as the first stage of a gesture understanding system.

[Note: The sequences from which Fig. 6 and 7 were taken are available at www.ifp.uiuc.edu/~jojic]

References:

[1] D. Metaxas, D. Terzopoulos, “Shape and nonrigid motion estimation through physics-based synthesis,” *IEEE Transactions on Pattern Analysis & Machine Intelligence, vol.15, no.6, pp.580-91, June 1993.*

[2] I. Kakadiaris, D. Metaxas, “Model-based estimation of 3D human motion with occlusion based on active multi-viewpoint selection,” *Proceedings 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.81-7, 1996.

[3] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland, “Pfinder: real-time tracking of the human body,” *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol.19, no.7, pp.780-5, July 1997.

[4] S. Ju, M. Black, Y. Yacoob, “Cardboard people: a parameterized model of articulated image motion,” *Proceedings of the Second International Conference on Automatic Face and Gesture*, pp.38-44, *1996.*

[5] A. Azarbayejani, A. Pentland, “Real-time self-calibrating stereo person tracking using 3-D shape estimation from blob features,” *Proceedings of the 13th International Conference on Pattern Recognition*, vol. 3, pp.627-32, 1996.

[6] C. Wren, A. Pentland, “Dynamic models of human motion,” *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, pp.22-7, 1998.

[7] C. Bregler, “Learning and recognizing human dynamics in video sequences,” *Proceedings. 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.568-74, 1997

[8] C. Eveland, K. Konolige, R. Bolles, “Background modeling for segmentation of video-rate stereo sequences,” *Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.266-71, 1998.

[9] B. Frey, *Graphical Models for Machine Learning and Digital Communications*, MIT Press, Cambridge MA., 1998.

[10] C. Bregler, J. Malik, “Tracking people with twists and exponential maps,” *Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern,* pp.8-15, 1998

[11] M. Yamamoto, A. Sato, S. Kawada, T. Kondo, Y. Osaki, “Incremental tracking of human actions from multiple views,” *Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* pp.2-7, 1998.

[12] O. Faugeras, *Three-Dimensional Computer Vision*, MIT Press, 1993.

[13] H. V. Poor, An Introduction to Signal Detection and Estimation, Springer-Verlag, 1994.

[14] T. Darrell, G. Gordon, M. Harville and J. Woodfill, “Integrated person tracking using stereo, color, and pattern detection,” *Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* pp. 601-8, 1998.

[15] J. Rehg and T. Kanade, “Model-based tracking of self-occluding articulated objects,” *Proceedings. 1995 International Conference on Computer Vision,* pp. 35-46, 1995.

[17] K. Konolige, Small Vision Systems: Hardware and Implementation *Eighth International Symposium on Robotics Research*, Hayama, Japan, October 1997.

[16] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm (with discussion),” *J. R. Statist. Soc. B* ***39***, pp. 1-38, 1977.

[17] K. Konolige, Small Vision Systems: Hardware and Implementation *Eighth International Symposium on Robotics Research*, Hayama, Japan, October 1997.