M.I.T Media Laboratory Perceptual Computing Section Technical Report No. 451
To appear: Journal of Image and Vision Computing special issue on Face and Gesture Recognition

DYNAMAN: A Recursive Model of Human Motion

Christopher R. Wren, Alex P. Pentland

MIT Media Laboratory; 20 Ames Street; Cambridge MA 02139 USA
http://www.media.mit.edu/vismod/ {cwren, sandy}@media.mit.edu
http://www.media.mit.edu/vismod/

Abstract

This paper describes a real-time, fully-dynamic, 3-D person tracking system that is able to tolerate full (temporary) occlusions and whose performance is substantially unaffected by the presence of multiple people. The framework provides a mathematically concise formulation for incorporating a wide variety of physical constraints and probabilistic influences. The framework takes the form of a recursive filter that enables pixel-level, probabilistic processes to take advantage of the contextual knowledge encoded in the higher-level models. Results are shown that demonstrate both qualitative and quantitative gains in tracking performance.

1 Introduction

This paper describes a real-time, fully-dynamic, 3-D person tracking system that is able to tolerate full (temporary) occlusions and whose performance is substantially unaffected by the presence of multiple people. The system is driven using 2-D *blob features* observed in two or more cameras [2, 22]. These features are then probabilistically integrated into a fully-dynamic 3-D skeletal model, which in turn drives the 2-D feature tracking process by providing 2-D projections of the 3-D model predicitions.

The feedback between 3-D model and 2-D image features is a recursive framework similar to an extended Kalman filter. Previous attempts at person tracking have utilized generic, mid-level image features (such as edges) that are computed as a preprocessing step, without consideration of the task to be accomplished. Our application is unusual, because the framework directly couples raw pixel measurements with an articulated, dynamic model of the human skeleton: no part of the system is blindly feed-forward. In this aspect our system is similar to that of Dickmanns in automobile control [6], and we obtain similar advantages in efficiency and stability though this direct coupling.

This paper will begin by describing our formulation for driving a 3-D skeletal model from 2-D probabilistic measurements, and then address the problem of incorporating feedback from the 3-D model to the 2-D feature finding process. Finally, we will report on experiments showing an increase in 3-D tracking accuracy, insensitivity to temporary occlusion, and the ability to handle multiple people.

1.1 Related Work

In recent years there has been much interest in tracking the human body using 3-D models with kinematic and dynamic constraints. Perhaps the first efforts in body tracking were by Badler and O'Rourke 1980[15], followed by Hogg 1988 [14], and other variations on their basic method[20, 3]. With the exception of Badler, these early efforts employed 2-D kinematic models of the human body driven by edge information.

Gavrila and Davis [8] and Rehg and Kanade [19] used 3-D kinematic models driven by edge data from multiple cameras in an analysis-synthesis framework. Both of these systems used the kinematic models to deal with limited occlusions, and thus could begin to handle a greater range of body motions.

In parallel to that work, some researchers began using dynamic models to track the human body. Pentland and Horowitz 1991 employed non-rigid finite element models driven by optical flow [16], and Metaxas and Terzopolous's 1993 system employing deformable superquadrics [10, 13] driven by 3-D point and 2-D edge measurements. More recently Bregler[5] has combined dynamic techniques with 2-D, region-based features with good results.

These systems all suffer from problems with self-occlusion. In general, they either rely on expensive search techniques to find consistent solutions or simply drop occluded body parts. This situation is exacerbated by the common use of the analysis-synthesis framework that forces early processing stages to make decisions without the benefit of context that is readily available in other parts of the system. Isard and Blake[9] call this the problem of ambiguous data. They suggest the Condensation method that avoids this search by using a probabilistic framework to carry a multitude of possible hypotheses. Unfortunately it requires the propagation hundreds to thousands of hypotheses to track even a single hand, and thus remains very far from real-time for this problem.

Recent work by Metaxas[11] comes close to breaking the barrier between ambiguous features and the context needed to resolve them by using dynamic predictions to drive a view-point selection algorithm, but the feature computation itself is a blind feed-forward process.

This work combines 3-D dynamic and kinematic models with region-based features to solve the body tracking problem. It is unique in two important ways: it is a fully recursive formulation and it is also completely real-time.

As we will explain below, this recursive framework makes the system robust to occlusions and numerous other visual ambiguities.

2 Mathematical Formulation

The human body is a complex dynamic system, whose visual features are time-varying, noisy signals. Accurately tracking the state of such a system requires use of a recursive estimation framework, as illustrated in figure 1. The elements of the framework are the observation model relating noisy low-level features to the higher-level skeletal model and *vice versa*, and the dynamic skeletal model itself. We will first describe the dynamic skeletal model, and then the observation model and its inverse. Finally we descibe a modification that addresses performance and engineering concerns.

2.1 Full Dynamics

There are a wide variety of ways to model physical systems. The model needs to include parameters that describe the *links* that compose the system, as well as information about the hard *constraints* that connect these links to one another. A model that only includes this information is called a *kinematic* model, and can only describe the static states of a system. The state vector of a kinematic model consists of the model state, $\mathbf{q}$.

A system in motion is more completely modeled when the *dynamics* of the system are modeled as well. A dynamic model describes the state evolution of the system over time. In a dynamic model the state vector includes velocity as well as position: $\mathbf{q}, \dot{\mathbf{q}}$. And state evolves according to Newton's First Law:

$$\ddot{\mathbf{q}} = \mathbf{W} \cdot \mathbf{Q}$$

Where $\mathbf{Q}$ is the vector of external forces applied to the system, and $\mathbf{W}$ is the inverse of the system mass matrix. The mass matrix describes the distribution of mass in the system.

2.1.1 Hard Constraints

Hard constraints represent absolute limitations imposed on the system. One example of a kinematic constraint is a skeletal joint. *Lagrangian dynamics* can be used to satisfy such hard constraints by choosing a state vector the describes only the degrees of freedom left to the system[7]. This requires that the constraints be solved analytically at model-building time.

While this formulation is very efficient, it requires an analytical solution of the constraints to exist. It also precludes modification of the constraint structure at run-time. Our model instead follows the *virtual work* formulation [21]. In a virtual work formulation, all the links in a model have full range of unconstrained motion. Hard kinematic constraints on the system are be represented as a special set of forces $\mathbf{C}$:

$$\ddot{\mathbf{q}} = \mathbf{W} \cdot (\mathbf{Q} + \mathbf{C}) \tag{1}$$

These constraints are functions of the systems state, and may also be time varying:

$$\mathbf{C} = \mathbf{c}(\mathbf{q}, t)$$

The constraints are defined so that they are satisfied when $\mathbf{c}(\mathbf{q}, t) = 0$. If the constraint is satisfied at time t_0, then it will remain satisfied for all time if $\dot{\mathbf{c}} = 0$ and $\ddot{\mathbf{c}} = 0$ also remain zero for all time $t > t_0$. Since $\mathbf{c}$ depends on q and t, differentiation gives us:

$$\dot{\mathbf{c}} = \frac{\partial \mathbf{c}}{\partial \mathbf{q}}\dot{\mathbf{q}} + \frac{\partial \mathbf{c}}{\partial t}$$

and another differentiation:

$$\ddot{\mathbf{c}} = \frac{\partial \mathbf{c}}{\partial \mathbf{q}}\ddot{\mathbf{q}} + \frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{q}}\dot{\mathbf{q}} + \frac{\partial^2 \mathbf{c}}{\partial t^2} \tag{2}$$

Setting $\ddot{\mathbf{c}} = 0$ and combining Equation 2 with Equation 1 gives us a system of linear equations that describes the set of $\mathbf{C}$ that are consistent with continued constraint satisfaction:

$$\frac{\partial \mathbf{c}}{\partial \mathbf{q}} W(\mathbf{Q} + \mathbf{C}) + \frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{q}}\dot{\mathbf{q}} + \frac{\partial^2 \mathbf{c}}{\partial t^2} = 0 \tag{3}$$

This system, in general, has many solutions. One method of chosing a solution is the concept of *virtual work*. Stated simply, virtual work requires that constraints never change the energy of the system; constraints must do no work. This can be accomplished by insuring that $\mathbf{C}\partial\mathbf{q} = 0$. Since $\partial\mathbf{q}$ is required to be in the null space of the constraint Jacobian:

$$\frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{q}} d\mathbf{q} = 0$$

Requiring the constraint forces to lie in the null space complement satisfies the virtual work requirement:

$$C = \boldsymbol{\lambda}\frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{q}} \tag{4}$$

Combining Equation 4 with Equation 3 results in an equation with only the vector of unknowns , $\boldsymbol{\lambda}$:

$$-\left[\frac{\partial \mathbf{c}}{\partial \mathbf{q}}^T \mathbf{W} \frac{\partial \mathbf{c}}{\partial \mathbf{q}}\right]\boldsymbol{\lambda} = \frac{\partial \mathbf{c}}{\partial \mathbf{q}}^T \mathbf{W}\mathbf{Q} + \frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{q}}\dot{\mathbf{q}} + \frac{\partial^2 \mathbf{c}}{\partial t^2}$$

This equation can be rewritten to emphasize its linear nature. $\mathbf{J}$ is the constraint Jacobian, $\boldsymbol{\kappa}$ is the vector on the right-hand side, and $\boldsymbol{\lambda}$ is the vector of unknown Lagrange multipliers:

$$-\mathbf{J}^T\mathbf{W}\mathbf{J}\boldsymbol{\lambda} = \boldsymbol{\kappa} \tag{5}$$

Biconjugate Gradient Decent The linear system in Equation 5 is large. Each rigid element in a 3-D model contributes 6 variables to the length of $\mathbf{q}$. Each constraint contributes a number of variable to the length of $\boldsymbol{\lambda}$ equal to the number of unconstrained degrees of freedom. So for a 5-link upper-body model, $\mathbf{W}$ is rank 30, and $\boldsymbol{\lambda}$ is length 15.

Fortunately the equation has some special structure. $\mathbf{W}$ is block diagonal and is therefore easy to compute from it's

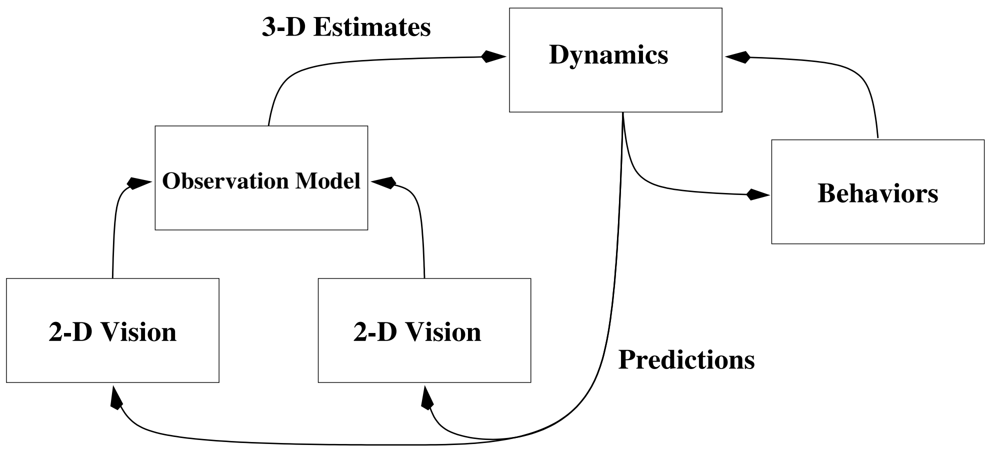

Figure 1: The flow of information though the system: 2-D observations to a 3-D dynamic model that feeds back to the 2-D observation process.

block diagonal inverse, the system mass matrix $\mathbf{M}$. The constraint Jacobian $\mathbf{J}$ is sparse when the number of constraints is $O(n)$, where n is the dimension of q.

The biconjugate gradient descent method for sparse matrix systems[18] takes advantage of this special structure to solve for $\boldsymbol{\lambda}$ stably and efficiently. The algorithm works by iteratively searching for the minimum of the potential field:

$$\mathcal{F}(x) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x}$$

This minimum occurs at the point where the gradient is zero:

$$\nabla\mathcal{F} = \mathbf{A}\mathbf{x} - \mathbf{b} = 0$$

For the constraint satisfaction problem $\mathbf{x}$ is the $\boldsymbol{\lambda}$ we wish to find, $\mathbf{A}$ is the sparse matrix $-\mathbf{J}^T\mathbf{W}\mathbf{J}$, and $\mathbf{b}$ is $\boldsymbol{\kappa}$ from Equation 5.

2.1.2 Soft Constraints

We must also integrate image observation data into the time evolution of the dynamic model. To accomplish this we use the observation model developed in our earlier stereo blob-tracking work (briefly described below), which relates the 2-D distribution of pixel values to a tracked object's 3-D position and orientation, thus providing a bridge to the physical modeling layer.

Because these observation data are noisy, they influence the dynamic model but do not impose hard constraints on its behavior. Consequently, observation data provide a sort of *soft constraint* on the model, and may be thought of as forces acting on the model's dynamics. For a large class of Kalman filters this analogy is exact, that is, the mathematics of integrating forces acting on a dynamic body is identical to the mathematics of observation integration within a Kalman filter [16, 13, 4]. Soft constraints have the additional advantage that they can also be used to model external influences on the dynamic system, such as gravity.

Soft constraints such as these can be expressed as a potential field acting on the dynamic system. A potential field is function over space that, when evaluated at a given position, applies a force to the model:

$$\mathbf{Q}_f = f(\mathbf{p})$$

Where $\mathbf{Q}_f$ is the component of $\mathbf{Q}$ contributed by the potential field when measured at point $\mathbf{p}$. The potential field is a good abstraction for modeling data since typical sensor noise distributions can be easily modeled as deformations of the field, for instance, measurement uncertainty in one direction can be modeled by making the potential field proportionally broader or narrower. The incorporation of a potential field function that models a probability density pushes the dynamic evolution of the model toward the most likely value, starting from the current model state.

Note that functions that take the model state as input, such as a control law, can also be represented as a time-varying potential field. One relevant example is incorporation of a probability distribution over link position and velocity:

$$\mathbf{Q}_f = f(\mathbf{p}, \mathbf{q}, \dot{\mathbf{q}})$$

2.2 The Observation Model

The low-level features extracted from video comprise the final element of our system. Our system tracks regions that are visually similar, and spatially coherent: blobs [17, 12]. The blob spatial statistics are described in terms of their second-order properties for which we denote $\boldsymbol{\mu}$ as the mean and $\boldsymbol{\Lambda}$ as the covariance. For computational convenience we will interpret this as a Gaussian model, so the probability of an observaion $\mathbf{o}$ (of a certain color and position within the image), given a blob model is:

$$\Pr(\mathbf{o}|\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k) = \frac{\exp(-\frac{1}{2}(\mathbf{o} - \boldsymbol{\mu}_k)^T\boldsymbol{\Lambda}_k^{-1}(\mathbf{o} - \boldsymbol{\mu}_k))}{(2\pi)^{\frac{m}{2}}|\boldsymbol{\Lambda}_k|^{\frac{1}{2}}}$$

The Gaussian interpretation is not terribly significant, because we also keep a pixel-by-pixel *support map* showing the actual occupancy [22].

Like other representations used in computer vision and signal analysis, including superquadrics, modal analysis, and eigen representations, blobs represent the global aspects of the shape and can be augmented with higher-order statistics to attain more detail if the data supports it. The reduction of degrees of freedom from individual pixels to blob parameters is a form of regularization which allows the ill-conditioned problem to be solved in a principled and stable way.

These 2-D features are the input to the 3-D blob estimation formulation used by Azarbayejani and Pentland [2]. This relates the 2-D distribution of pixel values to a tracked object's 3-D position and orientation. In our current system we track the hands and head using their color and shape characteristics; the observation equation therefore relates the distribution of hand/face pixel values to the probability distribution of the 3-D state variables that characterize the skeletal models' hand and head links.

These observations supply constraints on the underlying 3-D human model. Due to their probabilistic nature, observations are easily modeled as soft constraints. Observations are integrated into the dynamic evolution of the system by modeling them as descriptions of potential fields, as discussed in Section 2.1.2. The simplest such model is a linear spring model with a fixed *a priori* weighting ρ:

$$\mathbf{Q}_k = \rho\|\mathbf{q} - \boldsymbol{\mu}_k\|$$

A more sophisticated model takes into account the covariance of the observation model, $\boldsymbol{\Lambda}$, to create a more detailed potential field:

$$\mathbf{Q}_k = \rho(\mathbf{q} - \boldsymbol{\mu}_k)^T \boldsymbol{\Lambda}_k^{-1}(\mathbf{q} - \boldsymbol{\mu}_k)$$

Here ρ is again an *a priori* weighting of the influence of the observations on the model. Thus ρ plays the role of the Kalman gain in the recursive estimation process, and its value is set accordingly.

2.2.1 The Inverse Observation Model

It only remains to bring information from the human model back down to the initial stages of the vision system. In the absence of this information, the pixel classification decisions were forced to rely solely on temporal smoothness constraints in the 2-D image plane. The decision rule takes the form:

$$\Gamma_{ij} = \arg\max_k \left[\Pr(\mathbf{o}_{ij}|\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k)\right] \tag{6}$$

where Γ_{ij} is the labeling of pixel (i, j), and $(\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k)$ are the second-order statistics of model k.

Since the human model exists in 3-D a projection operation is required to convert the model's 3-D predictions into the 2-D features of the vision system. Given the current state of the model $\mathbf{q}$, it is possible to compute the state of an individual link that matches a specific tracked feature (say the hand), and compute 3-D means and covariances. Then, given a model of the camera, it is possible to calculate the projection of that state into 2-D and call it $(\boldsymbol{\mu}^*, \boldsymbol{\Lambda}^*)$. For the first moment (the mean) that calculation is a perspective projection:

$$\boldsymbol{\mu}^* = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \frac{x}{1+\frac{z}{f}} \\ \frac{y}{1+\frac{z}{f}} \end{bmatrix}$$

where (x, y, z) is the mean of the 3-D description of the link, f is the focal length of the camera model, and (u, v) is the projection of the mean into 2-D. Projection of the second moments is more difficult since the perspective projection of a Gaussian distribution is not itself Gaussian. We employ an approximation:

$$\boldsymbol{\Lambda}^* = \frac{\boldsymbol{\Lambda}_{xy}}{(1 + \frac{z}{f})^2}$$

where $\boldsymbol{\Lambda}_{xy}$ is the orthogonal projection of the 3-D covariance.

Since the vision system uses a stochastic framework, it is necessary to represent this link projection as a probabilistic model:

$$\Pr(\mathbf{o}|\mathbf{q}) = \frac{\exp(-\frac{1}{2}(\mathbf{o} - \boldsymbol{\mu}_k^*)^T {\boldsymbol{\Lambda}_k^*}^{-1}(\mathbf{o} - \boldsymbol{\mu}_k^*))}{(2\pi)|\boldsymbol{\Lambda}_k^*|^{\frac{1}{2}}}$$

Now that the 3-D model features are projected into the 2-D camera coordinates, they can be integrated into the 2-D probabilistic decision framework. This provides the Maximum A Posteriori decision rule with the much better prior information contained in the higher-level models.

2.3 A Dynamics Optimization

Large dynamic systems like those described in Section 2.1 are difficult to engineer. There are many parameters that must be chosen, and furthermore there are few guidelines available, so, often a great deal of expertise with dynamic simulation is required to realize a working system. This sort of system is also computationally rather costly: the system described above consumes nearly 100% of a 500MHz Alpha 21164 processor.

The above system can be easily converted to an energy-based kinematics engine as describe by Witkin [21], by dropping the integration over acceleration. Here are the update equations for the *dynamic* case using the synchronous Euler method:

$$\begin{aligned} \ddot{\mathbf{q}}_t &= \mathbf{W} \cdot \mathbf{Q} \\ \dot{\mathbf{q}}_t &= \dot{\mathbf{q}}_{t-1} + \Delta t \cdot \ddot{\mathbf{q}}_{t-1} \\ \mathbf{q}_t &= \mathbf{q}_{t-1} + \Delta t \cdot \dot{\mathbf{q}}_{t-1} \end{aligned}$$

and here are the modified equations for the *kinematic* case:

$$\begin{aligned} \ddot{\mathbf{q}}_t &= \mathbf{W} \cdot \mathbf{Q} \\ \dot{\mathbf{q}}_t &= \Delta t \cdot \ddot{\mathbf{q}}_{t-1} \\ \mathbf{q}_t &= \mathbf{q}_{t-1} + \Delta t \cdot \dot{\mathbf{q}}_{t-1} \end{aligned}$$

notice the change in calcuation of $\dot{\mathbf{q}}_t$. In this form the system retains all it's power to find minimal solutions to multiple constraints and continues to be a fertile ground for sensor fusion. However, it loses its grip on Newtonian reality, and with that, much of its predictive power.

Fortunately there are other ways of making predictions about dynamic systems, including Kalman filters. Linear Kalman filters are computationally efficient, and, while Kalman filters also have a large number of parameters that must be chosen, there is a mature literature on the subject [1]. The update equation for the linear Kalman filter is the matrix form of the dynamic update equations above:

$$\begin{bmatrix} \mathbf{q}_t \\ \dot{\mathbf{q}}_t \\ \ddot{\mathbf{q}}_t \end{bmatrix} = \mathbf{\Phi}_{t-1} \begin{bmatrix} \mathbf{q}_{t-1} \\ \dot{\mathbf{q}}_{t-1} \\ \ddot{\mathbf{q}}_{t-1} \end{bmatrix} + \mathbf{w}_{t-1} \qquad (7)$$

where $\mathbf{\Phi}_{t-1}$ is the system transition matrix. In our system $\mathbf{\Phi}$ is constant and represents a constant acceleration model Deviations from that model are captured in the random vector $\mathbf{w}$ that represents the system noise. Kalman filters don't require multiple calculations per observation to converge. At each observation time the optimal state estimate is calculated with the well known linear equation:

$$\begin{bmatrix} \hat{\mathbf{q}}_{t|t} \\ \hat{\dot{\mathbf{q}}}_{t|t} \\ \hat{\ddot{\mathbf{q}}}_{t|t} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{q}}_{t|t-1} \\ \hat{\dot{\mathbf{q}}}_{t|t-1} \\ \hat{\ddot{\mathbf{q}}}_{t|t-1} \end{bmatrix} + \mathbf{K}_t \left(\mathbf{y}_t - \mathbf{H}_t \begin{bmatrix} \hat{\mathbf{q}}_{t|t-1} \\ \hat{\dot{\mathbf{q}}}_{t|t-1} \\ \hat{\ddot{\mathbf{q}}}_{t|t-1} \end{bmatrix} \right)$$

Where $\mathbf{K}_t$ is the current Kalman Gain, and $\mathbf{y}_t$ is the observed 3-D position. $\mathbf{H}_t$ is the linear map between observation vector and the state vector. $\mathbf{H}_t$ can be linear since the perspective non-linearities are resolved by another part of the system as described in Section 2.2. $\mathbf{H}_t$ may be time-varying depending on the availability of observations at time t. The Kalman gain $\mathbf{K}_t$ is computed in the standard way, recursively calculating several internal covariance matrices including the error covariance estimate $\mathbf{P}_{t|t}$:

$$\mathbf{P}_{t|t} = (\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)\, \mathbf{P}_{t|t-1}$$

where $\mathbf{P}_{t|t-1}$ is the previous error covariance prediciton:

$$\mathbf{P}_{t|t-1} = \mathbf{\Phi}\mathbf{P}_{t-1|t-1}\mathbf{\Phi}^T + \mathbf{R}$$

and $\mathbf{R}$ is the covariance of $\mathbf{w}$ in Equation 7. $\mathbf{P}_{t|t-1}$ is a measure of our confidence in our predictions. The norm $\|\mathbf{P}_{t|t-1}\|$ is used as a confidence metric by the observation model described in Section 2.2.1. $\mathbf{P}_{t|t-1}$ is also used to update $\mathbf{K}_t$:

$$\mathbf{K}_t = \mathbf{P}_{t|t-1}\mathbf{H}_t^T \left[\mathbf{H}_t\mathbf{P}_{t|t-1}\mathbf{H}_t^T - \mathbf{R}\right]^{-1}$$

For a detailed derivation of these equations see Gelb [1].

A system that uses one such individual Kalman filter per tracked body part demonstrates good local tracking characteristics (in fact, optimal characteristics for some narrow definition of the word), but are subject to the failing that they occasionally make predictions that, taken together, violate the global, known constraints of the system. By linking independent Kalman filters together with the kinematics-only constraint formulation, we achieve qualitatively identical performance to the full-dynamics implementation with significantly less engineering and only 15% utilization of the same 500MHz Alpha 21164 processor. The state estimates of the Kalman filter simply become the goals for the constraint system. The kinematics iterate to a globally consistent answer, and those results then become the new state estimates.

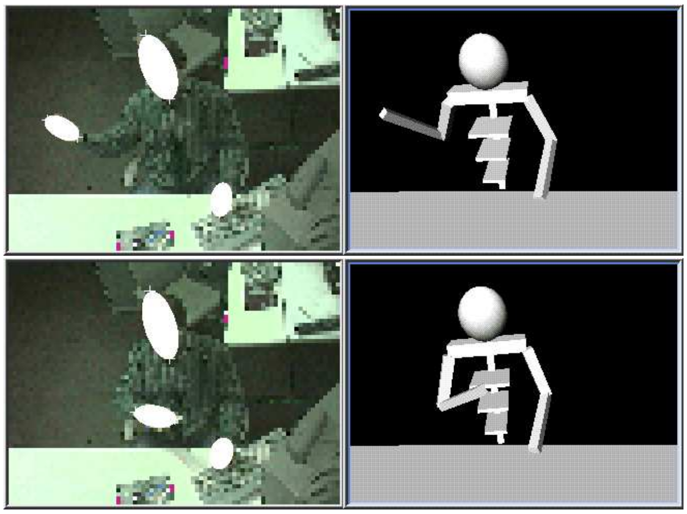

Figure 2: Frames on the left show video and 2-D blobs from one camera in the stereo pair. Frames on the right show corresponding configurations of the dynamic model at that instant in time.

A more intimate connection could be achieved with considerably more work and computation by using the Jacobians calculated in the constraint system to guide extended Kalman filters, but we didn't find this to be necessary.

It's important to remember that these linear Kalman filters sit *within* the larger recurive estimation framework described in the rest of this paper. They share the "Dynamics" box in Figure 1 with the constraint system.

3 Results

The dynamic skeleton model currently includes the upper body and arms. Figure 2 shows the real-time response to various target postures. The model interpolates those portions of the body state that are not measured directly, such as the upper body and elbow orientation, by use of the model's intrinsic dynamics and the behavior (control) models. The model also rejects noise that is inconsistent with the dynamic model. Table 1 compares RMS noise in the dynamic model output with noise in the underlying feature tracker. The "line following" test measures error from the best-fit line to data produced by constraining the users hand to move along a linear trajectory. The "rotational jitter" measures error to a smoothed version of data obtained by smooth motions of the user's hand through a rotation.

It can be seen that Figure 3 illustrates another advantage of feedback from higher-level models to the low-level vision system. Without feedback, the 2-D tracker fails if there is even partial self-occlusion from a single camera's perspective. With feedback, information from the dynamic model can be used to resolve ambiguity during 2-D tracking.

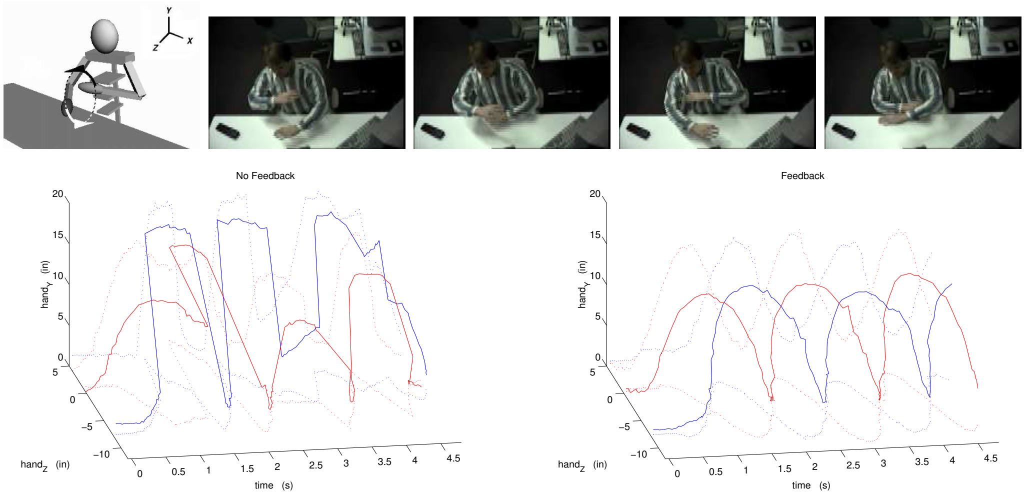

Figure 3: Tracking performance on a sequence with significant occlusion. **Top:** A diagram of the sequence and a single camera's view of the motion **Left:** This graph shows an example of tracking results without feedback. **Right:** This shows an example of correct tracking when feedback is enabled. Notice the smooth traces in the YT and ZT projections.

	tracker	dynamic model
line following	1.4 cm	0.9 cm
rotational jitter	2.2 deg	0.6 deg

Table 1: Comparison of RMS tracking error for tracking with and without dynamic feedback.

Figure 4: Users sharing the workspace. Physical constraints stabilize the 2-D tracker with respect to competing targets.

The model predictions also stabilize tracking by providing constraints that help the tracking algorithm reject distractions in the environment. The addition of another person to the scene, as in Figure 4, produces many patches in the image that are similar to the target blobs. Without high-level model knowledge, the 2-D tracker can only reject these distractions based on some assumptions about the temporal stability of blobs. With the addition of high-level feedback, however, the 2-D tracker now has information about the physical constraints of the underlying system. Consequently, it is generally not distracted by competing targets (such as other people).

The full system currently runs at at 30Hz utilizing four computers: 60% of an SGI 175MHz R10K O_2 per camera for blob-level processing, 100% of an Alpha 500MHz 21164 for dynamics and constraints, and 30% of an SGI 195MHz R10K Indigo^2 Impact for stereo processing and graphics rendering. The computers are connected to 100Mbps switched ethernet. The optimizations described in Section 2.3 reduce the utilization of the Alpha to 15%. That optimized dynamics and constraint module could be migrated to one of the under-utilized SGIs.

4 Conclusion

We have presented a real-time, fully-dynamic, multi-camera, 3-D person tracking system. The system adopts an extended Kalman filter framework that reconciles the 2-D tracking process with high-level 3-D models. This stabilizes the 2-D tracking by directly coupling an articulated dynamic model with raw pixel measurements. Some of the demonstrated benefits of this added stability include: increase in 3-D tracking accuracy, insensitivity to temporary occlusion, and the ability to handle multiple people.

5 References

[1] The Analytical Sciences Corporation. *Applied Optimal Estimation*, 1996.

[2] Ali Azarbayejani and Alex Pentland. Real-time self-calibrating stereo person tracking using 3-D shape estimation from blob features. In *Proceedings of 13th ICPR*, Vienna, Austria, August 1996. IEEE Computer Society Press.

[3] A. Baumberg and D. Hogg. An efficient method for contour tracking using active shape models. In *Proceeding of the Workshop on Motion of Nonrigid and Articulated Objects*. IEEE Computer Society, 1994.

[4] T. E. Boult, S. D. Fenster, and T. O. O'Donnell. Physics in a fantasy world vs. robust statistical estimation. In *Object Representation in Computer Vision. International NSF-ARPA Workshop*, pages 277–95, Berlin, Germany, 1995. Springer-Verlag.

[5] Christoph Bregler. Learning and recognizing human dynamics in video sequences. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, June 1997.

[6] Ernst D. Dickmanns and Birger D. Mysliwetz. Recursive 3-d road and relative ego-state recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(2):199–213, February 1992.

[7] Roy Featherstone. *Coordinate Systems and Efficiency*, chapter 8, pages 129–152. Kluwer Academic Publishers, 1984.

[8] D. M. Gavrila and L. S. Davis. Towards 3-d model-based tracking and recognition of human movement: a multi-view approach. In *International Workshop on Automatic Face- and Gesture-Recognition*. IEEE Computer Society, 1995. Zurich.

[9] Michael Isard and Andrew Blake. A mixed-state condensation tracker with automatic model-switching. In *Proc 6th Int. Conf. Computer Vision*, 1998.

[10] I. Kakadiaris, D. Metaxas, and R. Bajcsy. Active part-decomposition, shape and motion estimation of articulated objects: A physics-based approach. In *CVPR94*, pages 980–984, 1994.

[11] Ioannis Kakadiaris and Dimitris Metaxas. Vision-based animation of digital humans. In *Computer Animation*, pages 144–152. IEEE Computer Society Press, 1998.

[12] R. J. Kauth, A. P. Pentland, and G. S. Thomas. Blob: An unsupervised clustering approach to spatial preprocessing of mss imagery. In *11th Int'l Symposium on Remote Sensing of the Environment*, Ann Arbor, MI, April 1977.

[13] Dimitris Metaxas and Dimitris Terzopoulos. Shape and non-rigid motion estimation through physics-based synthesis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(6):580–591, 1993.

[14] K. Oatley, G. D. Sullivan, and D. Hogg. Drawing visual conclusions from analogy: preprocessing, cues and schemata in the perception of three dimensional objects. *Journal of Intelligent Systems*, 1(2):97–133, 1988.

[15] J. O'Rourke and N.I. Badler. Model-based image analysis of human motion using constraint propagation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2(6):522–536, November 1980.

[16] A. Pentland and B. Horowitz. Recovery of nonrigid motion and structure. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(7):730–742, July 1991.

[17] Alex Pentland. Classification by clustering. In *Proceedings of the Symposium on Machine Processing of Remotely Sensed Data*. IEEE, IEEE Computer Society Press, June 1976.

[18] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: the art of scientific computing*. Cambridge University Press, Cambridge, U.K., second edition, 1992.

[19] J.M. Rehg and T. Kanade. Visual tracking of high dof articulated structures: An application to human hand tracking. In *European Conference on Computer Vision*, pages B:35–46, 1994.

[20] K. Rohr. Cvgipiu. *"Towards Model-Based Recognition of Human Movements in Image Sequences*, 1(59):94–115, 1994.

[21] Andrew Witkin, Michael Gleicher, and William Welch. Interactive dynamics. In *ACM SIGGraph, Computer Graphics*, volume 24:2, pages 11–21. ACM SIGgraph, March 1990.

[22] Christopher Wren, Ali Azarbayejani, Trevor Darrell, and Alex Pentland. Pfinder: Real-time tracking of the human body. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(7):780–785, July 1997.