MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Perception for Human Motion Understanding

Christopher R. Wren, Ph.D.

TR2004-108 August 2004

Abstract

The fact that people are embodied places powerful constraints on their motion. By leveraging these constraints, we can build systems to perceive human motion that are fast and robust. More importantly, by understanding how these constraint systems relate to one another, and to the perceptual process itself, we can make progress toward building systems that interpret, not just capture, human motion.

To Appear in Innovations in Machine Intelligence & Robot Perception.

April 1998: portions published in the Third IEEE International Conference on Automatic Face and Gesture Recognition
March 2000: portions published in the Fourth IEEE International Conference on Automatic Face and Gesture Recognition
March 2000: portions published in Christopher R. Wren's Doctorial Thesis Dissertation for the Massachusetts Institute of Technology Electrical Engineering and Computer Science Department
August 2004: Invited chapter in Innovations in Machine Intelligence and Robot Perception.

The fact that people are embodied places powerful constraints on their motion. By leveraging these constraints, we can build systems to perceive human motion that are fast and robust. More importantly, by understanding how these constraint systems relate to one another, and to the perceptual process itself, we can make progress toward building systems that interpret, not just capture, human motion.

1 Overview

The laws of physics, the construction of the human skeleton, the layout of the musculature, the various levels of organization within the nervous system, the context of a task, and even forces of habits and culture all conspire to limit the possible configurations and trajectories of the human form. The kinematic constraints of the skeleton are instantaneous. They are always true (we hope), and serve to bound the domain of feasible estimates. The rest of these constraints exist to some degree in the temporal domain: given past observations, they tell us something about future observations.

These phenomenon cover a wide range of the time scales. The laws of physics apply in a continuous, instantaneous fashion. The subtle limits of muscle action may play out on time scales of milliseconds. Temporal structure due to the nervous system may range from tenths of seconds to minutes. Depending on the definition of a task, the task context may change over fractions of a minute or fractions of an hour. The subtle influence of affect might change over hours or even days. Habits and cultural norms develop over a lifetime.

A truly complete model of human embodiment would encompass all of these things. Unfortunately most of these phenomenon are currently beyond the scope of current modeling techniques. Neuroscience is only beginning to explain the impact of the structures of the peripheral nervous system on motion. Models of higher processes such as affect, task and culture are even farther away.

The things that we can model explicitly include the instantaneous geometric constraints (blobs, perspective, and kinematics) and the dynamic constraints of Newton's Laws. Blobs represent a visual constraint. We are composed of parts, and those parts appear in images as connected, visually coherent regions. Perspective constraints model the relationship between multiple views of the human body caused by our 3-D nature and the perspective projection of the world onto a CCD by a lens inside a camera. Kinematic constraints are the skeletal or connective constraints between the parts of the body: the length of a limb, the mechanics of a joint, and so on. The instantaneous configuration of the body is the *pose*. The kinematic constraints define the region of valid poses.

Newton's Laws represent a set of dynamic constraints: constraints in time. The assumption of bounded forces in the system implies bounded accelerations. Bounded accelerations in turn imply smoothness of the pose trajectory in time. Since the articulated frame of the body is complex and involves revolute joints, this isn't simply a smoothness constraint. It is a shaping function that is related to the global mass matrix which is a non-linear, time-varying function of the

pose.

The rest of the constraint layers (neuromuscular, contextual, and psychological) can currently only be modeled statistically through observation. Fortunately the recursive estimation framework discussed below offers a natural way to factor out these influences and treat them separately from the geometry and physics. Unfortunately, further factorization of the signal is a poorly understood problem. Since the signals occupy unknown, likely overlapping frequency bands, this leads to hard problems in system identification. As a result, we will treat these separate influences as a single, unified influence. This is obviously a simplification, but it is currently a necessary simplification.

1.1 Recursive Filters

The geometric constraints discussed above are useful for regularizing pose estimation, but the dynamic constraints provide something even more important: since they represent constraints in time, they allow prediction into the future. This is important because for human motion observed video rates, physics is a powerful predictor.

With a model of the observation process, predictions of 3-D body pose in the near future can be turned into predictions of observations. These predictions can be compared to actual observations when they are made. Measuring the discrepancy between prediction and observation provides useful information for updating the estimates of the pose. These differences are called innovations because they represent the aspects of the observations that were unpredicted by the model.

This link between model and observation is the powerful idea behind all recursive filters, including the well known Kalman filters. Kalman filters are the optimal recursive filter formulation for the class of problems with linear dynamics, linear mappings between state and observation, and white, Gaussian process noise. Extended Kalman filters generalize the basic formulation to include the case of analytically linearizable observation and dynamic models.

Recursive filters are able to cope with data in real time thanks to a Markovian assumption that the state of the system contains all the information needed to predict its behavior. For example, the state of a rigid physical object would include both the position and velocity. There is no need for the filter to simultaneously consider all the observations ever made of the subject to determine it's state. The update of the state estimate only requires combining the innovation with the dynamic, observation, and noise models.

The complete recursive loop includes measurement, comparison of predicted observation to actual observation, corresponding update of state estimate, prediction of future state estimate, and rendering of the next predicted observation. This is the basic flow of information in a Kalman filter, and applies equally well to recursive filters in general.

For the case of observing the human body, this general framework is complicated by the fact that the human body is a 3-D articulated system and the observation process is significantly non-trivial. Video images of the human body

are high-dimensional signals and the mapping between body pose and image observation involves perspective projection. These unique challenges go beyond the original design goals of the Kalman and extended Kalman filters and they make the task of building systems to observe human motion quite difficult.

1.2 Feedback for Early Vision

Most computer vision systems are modularized to help reduce software complexity, manage bandwidth, and improve performance. Often, low-level modules, comprised of filter-based pattern recognition pipelines, provide features to mid-level modules that then use statistical or logical techniques to infer meaning. The mid-level processes are made tractable by the dimensionality reduction accomplished by the low-level modules, but these improvements can incur a cost in robustness. These systems are often brittle: they fail when the assumptions in a low-level filter are violated. Once a low-level module fails, the information is lost. Even in the case where the mid-level module can employ complex models to detect the failure, there is no way to recover the lost information if there is no downward flow of information that can be used to avert the failure. The system is forced to rely on complex heuristics to attempt approximate repair[44].

Dynamic constraints enable the prediction of observations in the near future. These predictions, with the proper representation, can be employed by low-level perceptual processes to resolve ambiguities. This results in a more robust system, by enabling complex, high-level models to inform the earliest stages of processing. It is possible to retain the advantages of modularity in a closed-loop system through carefully designed interfaces.

For example, the DYNA system[52] measures the 2-D locations of body parts in the image plane using an image-region tracker. The system then estimates 3-D body part locations from stereo pairs of 2-D observations. Finally the full body pose is estimated from these 3-D observations using a 3-D, non-linear model of the kinematics and dynamics of the human body. This system is well modularized and fast, but but would be very brittle if it relied on information only flowing from low-level processes to high-level interpretation. Instead, predictions from the dynamic model are incorporated as prior information into the probabilistic blob tracker. The tracker is the first process to be applied to the pixels, so given this feedback, there is no part of the system that is bottom-up. Even this lowest level pixel-classification process incorporates high-level model influence in the form of state predictions represented as prior probabilities for pixel classification.

This influence is more significant than simply modifying or bounding a search routine. Our classifier actually produces different results in the presence of feedback: results that reflect global classification decisions instead of locally optimal decisions that may be misleading or incomplete in the global context. This modification is made possible due to the statistical nature of our blob tracker. Prior information generated by the body model transforms the bottom-up, maximum likelihood blob tracker into a maximum *a posteriori* classifier. Thanks to the probabilistic nature of the blob tracker, it is possible to the hide

the details of the high-level processes from the low-level processes, and thereby retain the speed and simplicity of the pixel classification.

1.3 Expression

An appropriate model of embodiment allows a perceptual system to separate the necessary aspects of motion from the purposeful aspects of motion. The necessary aspects are a result of physics and are predictable. The purposeful aspects are the direct result of a person attempting to express themselves through the motion of their bodies. Understanding embodiment is the key to perceiving expressive motion.

Human-computer interfaces make measurements of a human and use those measurements to give them control over some abstract domain. The sophistication of these measurements range from the trivial keyclick to the most advanced perceptual interface system. Once the measurements are acquired the system usually attempts to extract some set of features as the first step in a pattern recognition system that will convert those measurements into whatever domain of control the application provides. Those features are usually chosen for mathematical convenience or to satisfy an *ad hoc* notion of invariance.

The innovations process discussed above is a fertile source of features that are directly related to the embodiment of the human. When neuromuscular, contextual or psychological influences affect the motion of the body, these effects will appear in the innovations process if they are not explicitly modeled. This provides direct access for learning mechanisms to these influences without compounding them with the effects of physics, kinematics, imaging, or any other process that can be explicitly modeled by the system. This tight coupling between appearance, motion and behavior is a powerful implication of this framework.

2 Theoretic Foundations

This section will expand on the ideas presented in Section 1, while linking them to their roots in stochastic estimation theory.

The fundamental idea presented in Section 1 is that perception is improved when it is coupled with expectations about the process being observed: specifically a model with the ability to make qualified predictions into the future given past observations. A logical framework for creating and employing this kind of model in a perceptual system can be found in the control and estimation literature. Since the human body is a physical system, it shares many properties with the general class of dynamic systems. It is instructive to approach the task of understanding human motion in the same way that an engineer might approach the task of observing any dynamic system.

One possible simplified block diagram of a human is illustrated in Figure 1. The passive, physical reality of the human body is represented by the *Plant*. The propagation of the system forward in time is governed by the laws of physics

and is influenced by signals, $\mathbf{u}$, from *Control*. On the right, noisy observations, $\mathbf{y}$, can be made of the Plant. On the left, high level goals, $\mathbf{v}$, are supplied to the Controller.

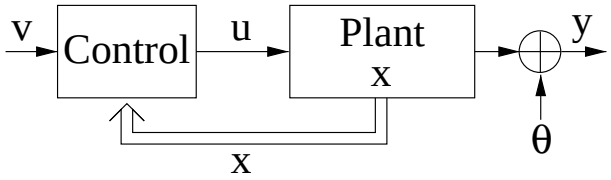

Figure 1: A systems view of the human body.

The observations are a function of the system state according to some measurement process, $h(\cdot)$. In our case this measurement process corresponds to the imaging process of a camera. As such, it is a non-linear, incomplete transform: cameras do not directly measure velocity, they are subject to occlusion, and they project 3-D information into 2-D observations:

$$\mathbf{y}_{t+1} = h(\mathbf{x}_{t+1}) + \boldsymbol{\theta}_t \tag{1}$$

The measurement process is also noisy. $\boldsymbol{\theta}_t$ represents additive noise in the observation. The $\boldsymbol{\theta}_t$ are assumed to be samples from a white, Gaussian, zero-mean process with covariance $\boldsymbol{\Theta}$:

$$\boldsymbol{\theta}_t \leftarrow \mathcal{N}(\mathbf{0}, \boldsymbol{\Theta}) \tag{2}$$

The state vector, $\mathbf{x}$, completely defines the configuration of the system in phase-space. The Plant propagates the state forward in time according to the system constraints. In the case of the human body this includes the non-linear constraints of kinematics as well as the constraints of physics. The Plant also reacts to the influences of the control signal. For the human body these influences come as muscle forces. It is assumed that the Plant can be represented by an unknown, non-linear function $f(\cdot, \cdot)$:

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t) \tag{3}$$

The control signals are physical signals, for example, muscle activations that result in forces being applied to the body. The Controller obviously represents a significant amount of complexity: muscle activity, the properties of motor nerves, and all the complex motor control structures from the spinal cord up into the cerebellum. The Controller has access to the state of the Plant, by the process of proprioception:

$$\mathbf{u}_t = c(\mathbf{v}_t, \mathbf{x}_t) \tag{4}$$

The high-level goals, $\mathbf{v}$, are very high-level processes. These signals represent the place where intentionality enters into the system. If we are building a system to interact with a human, then we get the observations, $\mathbf{y}$, and what we're really interested in is the intentionality encoded in $\mathbf{v}$. Everything else is just in the way.

2.1 A Classic Observer

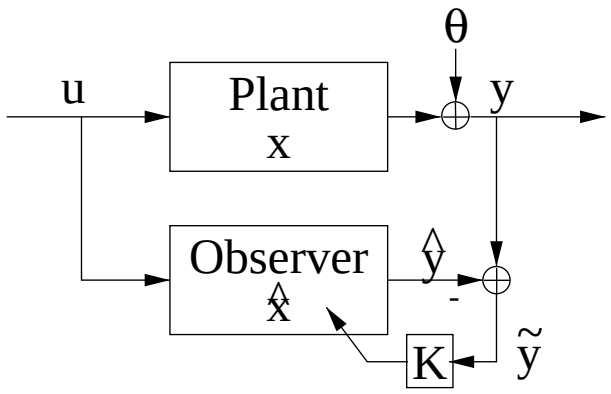

Figure 2: Classic observer architecture

A classic observer for such a system takes the form illustrated in Figure 2. This is the underlying structure of recursive estimators, including the well known Kalman and extended Kalman filters.

The *Observer* is an analytical model of the physical Plant:

$$\mathbf{x}_{t+1} = \mathbf{\Phi}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{u}_t + \mathbf{L}_t \boldsymbol{\xi}_t \tag{5}$$

The unknown, non-linear update equation, $f(\cdot,\cdot)$ from Equation 3, is modeled as the sum of two non-linear functions: $\Phi(\cdot)$ and $B(\cdot)$. $\Phi(\cdot)$ propagates the current state forward in time, and $B(\cdot)$ maps control signals into state influences. $\mathbf{\Phi}_t$ and $\mathbf{B}_t$ from Equation 5 are linearizations of $\Phi(\cdot)$ and $B(\cdot)$ respectively, at the current operating point. The right-hand term, $(\mathbf{L}_t \boldsymbol{\xi}_t)$, represents the effect of noise introduced by modeling errors on the state update. The $\boldsymbol{\xi}_t$ are assumed to be samples from a white, Gaussian, zero-mean process with covariance $\mathbf{\Xi}$ that is independent of the observation noise from Equation 2:

$$\boldsymbol{\xi}_t \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{\Xi}) \tag{6}$$

The model of the measurement process is also linearized. $\mathbf{H}_t$ is a linearization of the non-linear measurement function $h(\cdot)$:

$$\mathbf{y}_{t+1} = \mathbf{H}_t \mathbf{x}_{t+1} + \boldsymbol{\theta}_t \tag{7}$$

Linearizations, such as those invoked to form $\mathbf{\Phi}_t$ and $\mathbf{H}_t$, are performed by computing the Jacobian matrix. The Jacobian of a multivariate function of $\mathbf{x}$ such as $\mathbf{\Phi}(\cdot)$ is computed as the matrix of partial derivatives at the operating

point $\mathbf{x}_t$ with respect to the components of $\mathbf{x}$:

$$\boldsymbol{\Phi}_t = \nabla_{\mathbf{x}_\mathbf{x}} \boldsymbol{\Phi}|_{\mathbf{x}=\mathbf{x}_t} = \begin{bmatrix} \frac{\partial \boldsymbol{\Phi}_1}{\partial \mathbf{x}_1}\Big|_{\mathbf{x}=\mathbf{x}_t} & \frac{\partial \boldsymbol{\Phi}_1}{\partial \mathbf{x}_2}\Big|_{\mathbf{x}=\mathbf{x}_t} & \cdots & \frac{\partial \boldsymbol{\Phi}_1}{\partial \mathbf{x}_n}\Big|_{\mathbf{x}=\mathbf{x}_t} \\ \frac{\partial \boldsymbol{\Phi}_2}{\partial \mathbf{x}_1}\Big|_{\mathbf{x}=\mathbf{x}_t} & \ddots & & \vdots \\ \vdots & & \ddots & \\ \frac{\partial \boldsymbol{\Phi}_m}{\partial \mathbf{x}_1}\Big|_{\mathbf{x}=\mathbf{x}_t} & \cdots & \cdots & \frac{\partial \boldsymbol{\Phi}_m}{\partial \mathbf{x}_n}\Big|_{\mathbf{x}=\mathbf{x}_t} \end{bmatrix}$$

This operation is often non-trivial.

Estimation begins from a prior estimate of state: $\hat{\mathbf{x}}_{0|0}$. Given the current estimate of system state, $\hat{\mathbf{x}}_{t|t}$, and the update Equation 5, it is possible to compute a prediction for the state at $t+1$:

$$\hat{\mathbf{x}}_{t+1|t} = \boldsymbol{\Phi}_t \hat{\mathbf{x}}_{t|t} + \mathbf{B}_t \mathbf{u}_t \tag{8}$$

Notice that $\boldsymbol{\xi}_t$ falls out since:

$$E[\boldsymbol{\xi}_t] = E\left[\mathcal{N}(\mathbf{0}, \boldsymbol{\Xi})\right] = \mathbf{0} \tag{9}$$

Combining this state prediction with the measurement model provides a prediction of the next measurement:

$$\hat{\mathbf{y}}_{t+1|t} = \mathbf{H}_t \hat{\mathbf{x}}_{t+1|t} \tag{10}$$

Again, $\boldsymbol{\theta}_t$ drops out since:

$$E[\boldsymbol{\theta}_t] = E\left[\mathcal{N}(\mathbf{0}, \boldsymbol{\Theta})\right] = \mathbf{0} \tag{11}$$

Given this prediction it is possible to compute the residual error between the prediction and the actual new observation $\mathbf{y}_{t+1}$:

$$\tilde{\mathbf{y}}_{t+1} = \boldsymbol{\nu}_{t+1} = \mathbf{y}_{t+1} - \hat{\mathbf{y}}_{t+1|t} \tag{12}$$

This residual, called the innovation, is the information about the actual state of the system that the filter was unable to predict, plus noise. A weighted version of this residual is used to revise the new state estimate for time $t+1$ to reflect the new information in the most recent observation:

$$\hat{\mathbf{x}}_{t+1|t+1} = \hat{\mathbf{x}}_{t+1|t} + \mathbf{K}_{t+1} \tilde{\mathbf{y}}_{t+1} \tag{13}$$

In the case of the Kalman filter, the weighting matrix is the well-known Kalman gain matrix. It is computed from the estimated error covariance of the state prediction, the measurement models, and the measurement noise covariance, $\boldsymbol{\Theta}$:

$$\mathbf{K}_{t+1} = \boldsymbol{\Sigma}_{t+1|t} \mathbf{H}_t^T \left[\mathbf{H}_t \boldsymbol{\Sigma}_{t+1|t} \mathbf{H}_t^T + \boldsymbol{\Theta}_{t+1}\right]^{-1} \tag{14}$$

The estimated error covariance of the state prediction is initialized with the estimated error covariance of the prior state estimate, $\boldsymbol{\Sigma}_{0|0}$. As part of the

state prediction process, the error covariance of the state prediction can be computed from the error covariance of the previous state estimate using the dynamic update rule from Equation 5:

$$\mathbf{\Sigma}_{t+1|t} = \mathbf{\Phi}_t \mathbf{\Sigma}_{t|t} \mathbf{\Phi}_t^T + \mathbf{L}_t \mathbf{\Xi}_t \mathbf{L}_t^T \tag{15}$$

Notice that, since $\mathbf{u}_t$ is assumed to be deterministic, it does not contribute to this equation.

Incorporating new information from measurements into the system reduces the error covariance of the state estimate: after a new observation, the state estimate should be closer to the true state:

$$\mathbf{\Sigma}_{t+1|t+1} = \left[\mathbf{I} - \mathbf{K}_{t+1}\mathbf{H}_t\right] \mathbf{\Sigma}_{t+1|t} \tag{16}$$

Notice, in Equation 5, that that classic Observer assumes access to the control signal $\mathbf{u}$. For people, remember that the control signals represent muscle activations that are unavailable to a non-invasive Observer. That means that an observer of the human body is in the slightly different situation illustrated in Figure 3.

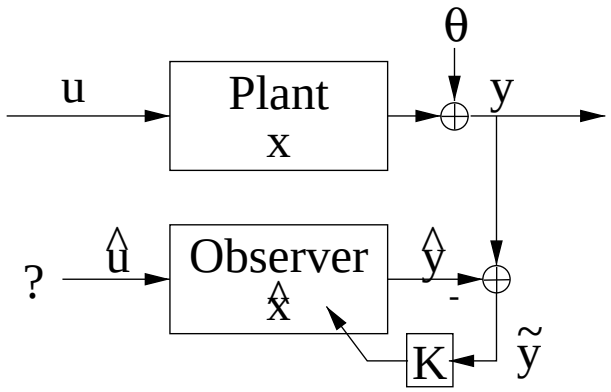

Figure 3: An Observer of the human body can't access $\mathbf{u}$

2.2 A Lack of Control

Simply ignoring the $(\mathbf{B}_t\mathbf{u})$ term in Equation 5 results in poor estimation performance. Specifically, the update Equation 13 expands to:

$$\hat{\mathbf{x}}_{t+1|t+1} = \hat{\mathbf{x}}_{t+1|t} + \mathbf{K}_{t+1}(\mathbf{y}_{t+1} - \mathbf{H}_t(\mathbf{\Phi}_t \hat{\mathbf{x}}_{t|t} + \mathbf{B}_t \mathbf{u}_t)) \tag{17}$$

In the absence of access to the control signal $\mathbf{u}$, the update equation becomes:

$$\tilde{\hat{\mathbf{x}}}_{t+1|t+1} = \hat{\mathbf{x}}_{t+1|t} + \mathbf{K}_{t+1}(\mathbf{y}_{t+1} - \mathbf{H}_t(\mathbf{\Phi}_t \hat{\mathbf{x}}_{t|t} + \mathbf{B}_t \mathbf{0})) \tag{18}$$

The error ε between the ideal update and the update without access to the control signal is then:

$$\begin{aligned} \varepsilon &= \left| \hat{\mathbf{x}}_{t+1|t+1} - \tilde{\hat{\mathbf{x}}}_{t+1|t+1} \right| && (19) \\ &= \mathbf{K}_{t+1}\mathbf{H}_t\mathbf{B}_t\mathbf{u}_t && (20) \end{aligned}$$

Treating the control signal, $\mathbf{u}_t$, as a random variable, we compute the control mean and covariance matrix:

$$\begin{aligned} \bar{\mathbf{u}} &= E[\mathbf{u}_t] && (21) \\ \mathbf{U} &= E[(\mathbf{u}_t - \bar{\mathbf{u}})(\mathbf{u}_t - \bar{\mathbf{u}})^T] && (22) \end{aligned}$$

If the control covariance matrix is small relative to the model and observation noise, by which we mean:

$$\begin{aligned} \|\mathbf{U}\| &<< \|\mathbf{\Xi}_t\| && (23) \\ \|\mathbf{U}\| &<< \|\mathbf{\Theta}_t\| && (24) \end{aligned}$$

then the standard recursive filtering algorithms should be robust enough to generate good state and covariance estimates. However, as $\mathbf{U}$ grows, so will the error ε. For large enough $\mathbf{U}$ it will not be possible to hide these errors within the assumptions of white, Gaussian process noise, and filter performance will significantly degrade [3].

It should be obvious that we expect $\mathbf{U}$ to be large: if $\mathbf{u}$ had only negligible impact on the evolution of $\mathbf{x}$, then the human body wouldn't be very effective. The motion of the human body is influenced to a large degree by the actions of muscles and the control structures driving those muscles. This situation will be illustrated in Section 3.

2.3 Estimation of Control

It is not possible to measure $\mathbf{u}_t$ directly. It is inadvisable to ignore the effects of active control, as shown above. An alternative is to estimate $\mathbf{u}_{t+1|t}$. This alternative is illustrated in Figure 4: assuming that there is some amount of structure in $\mathbf{u}$, the function $g(\cdot,\cdot)$ uses $\hat{\mathbf{x}}$ and $\tilde{\mathbf{y}}$ to estimate $\hat{\mathbf{u}}$.

The measurement residual, $\tilde{\mathbf{y}}_{t+1}$ is a good place to find information about $\mathbf{u}_t$ for several reasons. Normally, in a steady-state observer, the measurement residual is expected to be zero-mean, white noise, so $E[\tilde{\mathbf{y}}_t] = 0$. From Equation 20 we see that without knowledge of $\mathbf{u}_t$, $\tilde{\mathbf{y}}_{t+1}$ will be biased:

$$E[\tilde{\mathbf{y}}_{t+1}] = \mathbf{H}_t\mathbf{B}_t\mathbf{u}_t \quad (25)$$

This bias is caused by the faulty state prediction resulting in a biased measurement prediction. Not only will $\tilde{\mathbf{y}}_{t+1}$ not be zero-mean, it will also not be white. Time correlation in the control signal will introduce time correlation in the residual signal due to the slow moving bias. Specific examples of such structure in the residuals will be shown in Section 3.

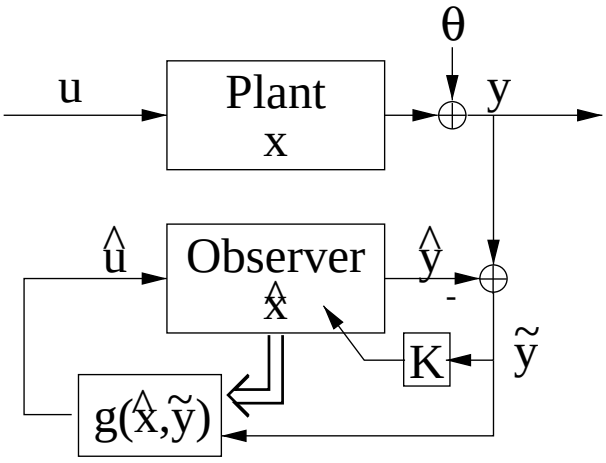

Figure 4: An observer that estimates $\hat{\mathbf{u}}$ as well as $\hat{\mathbf{x}}$

Learning the bias and temporal structure of the measurement residuals provides a mechanism for learning models of $\mathbf{u}$. Good estimates of $\mathbf{u}$ will lead to better estimates of $\mathbf{x}$ which are useful for a wide variety of applications including motion capture for animation, direct manipulation of virtual environments, video compositing, diagnosis of motor disorders, and others. However, if we remain focused on the intentionality represented by $\mathbf{v}$ on the far left of Figure 1, then this improved tracking data is only of tangential interest as a means to compute $\hat{\mathbf{v}}$.

The neuroscience literature[43] is our only source of good information about the control structures of the human body, and therefore the structure of $\mathbf{v}$. This literature seems to indicate that the body is controlled by the setting of goal states. The muscles change activation in response to these goals, and the limb passively evolves to the new equilibrium point. The time scale of these mechanisms seem to be on the scale of hundreds of milliseconds.

Given this apparent structure of $\mathbf{v}$, we expect that the internal structure of $g(\cdot,\cdot)$ should contain states that represent switches between control paradigms, and thus switches in the high-level intentionality encoded in $\mathbf{v}$. Section 3 discusses possible representations for $g(\cdot,\cdot)$ and Section 4 discusses results obtained in controlled contexts (where the richness of $\mathbf{v}$ is kept manageable by the introduction of a constrained context).

2.4 Images as Observations

There is one final theoretic complication with this formulation of an observer for human motion. Recursive filtering matured under the assumption that the measurement process produced low-dimensional signals under a measurement model that could be readily linearized: such as the case of a radar tracking a

ballistic missile. Images of the human body taken from a video stream do not fit this assumption: they are high dimensional signals and the imaging process is complex.

One solution, borrowed from the pattern recognition literature, is to place a deterministic filter between the raw images and the Observer. The measurements available to the Observer are then low-dimensional features generated by this filter [47]. This situation is illustrated in Figure 5.

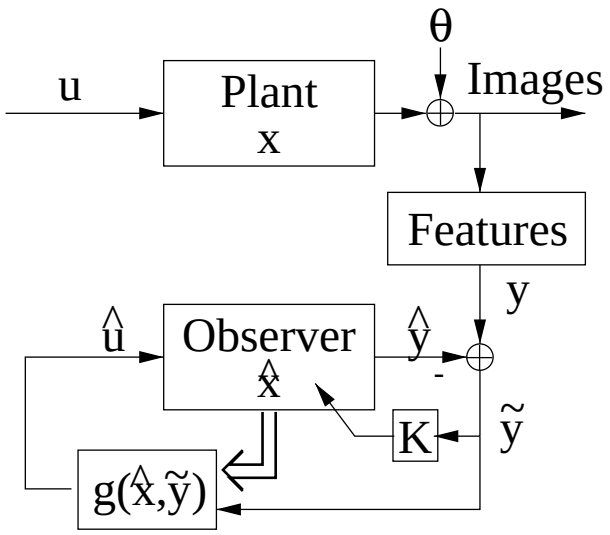

Figure 5: Feature Extraction between image observations and the Observer.

One fatal flaw in this framework is the assumption that it is possible to create a stationary filter process that is robust and able to provide all the relevant information from the image as a low dimensional signal for the Observer. This assumption essentially presumes a pre-existing solution to the perception problem. A sub-optimal filter will succumb to the problem of perceptual aliasing under a certain set of circumstances specific to that filter. In these situations the measurements supplied to the Observer will be flawed. The filter will have failed to capture critical information in the low-dimensional measurements. It is unlikely that catastrophic failures in feature extraction will produce errors that fit within the assumed white, Gaussian, zero-mean measurement noise model. Worse, the situation in Figure 5 provides no way for the predictions available in the Observer to avert these failures. This problem will be demonstrated in more detail in Section 5.

A more robust solution is illustrated in Figure 6. A steerable feature extraction process takes advantage of observation predictions to resolve ambiguities. It is even possible to compute an estimate of the observation prediction error covariance, $(\mathbf{H}_t \mathbf{\Sigma}_{t+1|t} \mathbf{H}_t^T)$ and weight the influence of these predictions according to their certainty. Since this process takes advantage of the available predictions

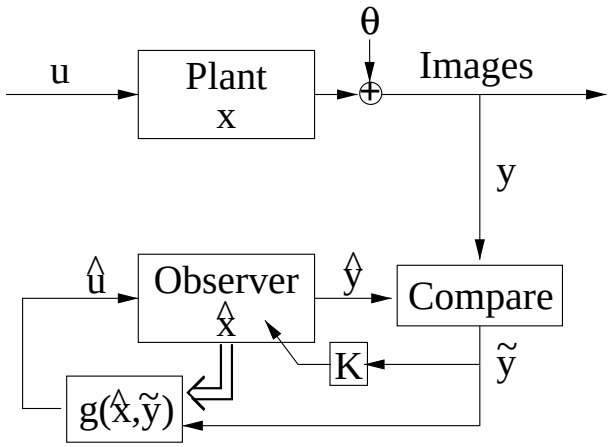

Figure 6: The Observer driving a steerable feature extractor.

it does not suffer from the problems described above, because prior knowledge of ambiguities enables the filter to anticipate catastrophic failures. This allows the filter to more accurately identify failures and correctly propagate uncertainty, or even change modes to better handle the ambiguity. A fast, robust implementation of such a system is described in detail in Section 3.

2.5 Summary

So we see that exploring the task of observing the human from the vantage of classical control theory provides interesting insights. The powerful recursive link between model and observation will allow us to build robust and fast systems. Lack of access to control signals represent a major difference between observing built systems and observing biological systems. Finally that there is a possibility of leveraging the framework to help in the estimation of these unavailable but important signals.

For the case of observing the human body, this general framework is complicated by the fact that the human body is a 3-D articulated system and the observation process is significantly non-trivial. Video images of the human body are extremely high-dimensional signals and the mapping between body pose and image observation involves perspective projection. These unique challenges go beyond the original design goals of the Kalman and extended Kalman filters and they make the task of building systems to observe human motion quite difficult. The details involved in extending the basic framework to this more complex domain are the subject of the next section.

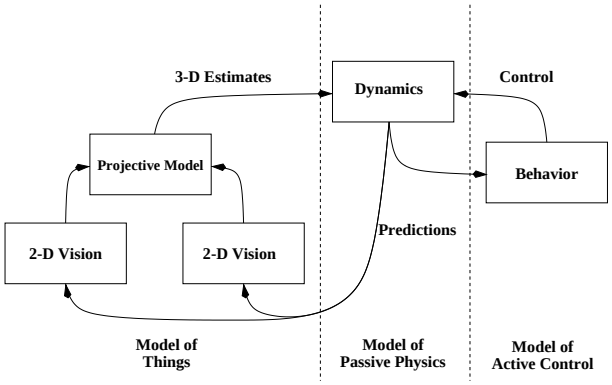

Figure 7: The Recursive Filtering framework. Predictive feedback from the 3-D dynamic model becomes prior knowledge for the 2-D observations process. Predicted control allows for more accurate predictive feedback.

3 An Implementation

This section attempts to make the theoretical findings of the previous section more concrete by describing a real implementation. The DYNA architecture is a real-time, recursive, 3-D person tracking system. The system is driven by 2-D *blob features* observed in two or more cameras [4, 53]. These features are then probabilistically integrated into a dynamic 3-D skeletal model, which in turn drives the 2-D feature tracking process by setting appropriate prior probabilities.

The feedback between 3-D model and 2-D image features is in the form of a recursive filter, as described in the previous section. One important aspect of the DYNA architecture is that the filter directly couples raw pixel measurements with an articulated dynamic model of the human skeleton. In this aspect the system is similar to that of Dickmanns in automobile control [15], and results show that the system realizes similar efficiency and stability advantages in the human motion perception domain.

This framework can be applied beyond passive physics by incorporating various patterns of control (which we call 'behaviors') that are *learned* from observing humans while they perform various tasks. Behaviors are defined as those aspects of the motion that cannot be explained solely by passive physics or the process of image production. In the untrained tracker these manifest as significant structures in the innovations process (the sequence of prediction errors). Learned models of this structure can be used to recognize and predict this purposeful aspect of human motion.

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 7. The framework consists of several modules. Section 3.1 details the module labeled

"2-D Vision". The module labeled "Projective Model" is described in [4] and is summarized. The formulation of our 3-D skeletal physics model, "Dynamics" in the diagram, is explained in Section 3.2, including an explanation of how to drive that model from the observed measurements. The generation of prior information for the "2-D Vision" module from the model state estimated in the "Dynamics" module is covered in Section 3.3. Section 3.4 explains the behavior system and its intimate relationship with the physical model.

3.1 The Observation Model

Our system tracks regions that are visually similar, and spatially coherent: blobs. We can represent these 2-D regions by their low-order statistics. This compact model allows fast, robust classification of image regions.

Given a pair of calibrated cameras, pairs of 2-D blob parameters are used to estimate the parameters of 3-D blobs that exist behind these observations. Since the stereo estimation is occurring at the blob level instead of the pixel level, it is fast and robust.

This section describes these low-level observation and estimation processes in detail.

3.1.1 Blob Observations

Clusters of 2-D points, (i, j), have 2-D spatial means and covariance matrices, which we shall denote $\boldsymbol{\mu}_s$ and $\boldsymbol{\Sigma}_s$. The blob spatial statistics are described in terms of these second-order properties. For computational convenience we will interpret this as a Gaussian model. The Gaussian interpretation is not terribly significant, because we also keep a pixel-by-pixel *support map* showing the actual blob occupancy in the video frame [51].

The visual appearance of the pixels, (y, u, v), that comprise a blob can also be modeled by second order statistics in color space: the 3-D mean, $\boldsymbol{\mu}_c$ and covariance, $\boldsymbol{\Sigma}_c$. As with the spatial statistics, these chromatic statistics are interpreted as the parameters of a Gaussian distribution in color space. We chose the YUV representation of color due to its ready availability from video digitization hardware and the fact that in the presence of white luminants it confines much of the effect of shadows to the single coordinate y [51].

Given these two sets of statistics describing the blob, the overall blob description becomes the concatenation $(ijyuv)$, where the overall mean is:

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_s \\ \boldsymbol{\mu}_c \end{bmatrix}$$

and the overall covariance is:

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_s & \boldsymbol{\Lambda}_{sc} \\ \boldsymbol{\Lambda}_{cs} & \boldsymbol{\Sigma}_c \end{bmatrix}$$

This framework allows for the concatenation of additional statistics that may be available from image analysis, such as texture or motion components. Figure 8

Figure 8: A person interpreted as a set of blobs.

shows a person represented as a set of blobs. Spatial mean and covariance is represented by the iso-probability contour ellipse shape. The color mean is represented by the color of the blob. The color covariance is not represented in this illustration.

3.1.2 Frame Interpretation

To compute $\Pr(\mathbf{O}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, the probability that a given pixel observation, $\mathbf{O}$, is a member of a given blob, k, we employ the Gaussian assumption to arrive at the formula:

$$\Pr(\mathbf{O}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{\exp(-\frac{1}{2}(\mathbf{O} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(\mathbf{O} - \boldsymbol{\mu}_k))}{(2\pi)^{\frac{m}{2}} |\boldsymbol{\Sigma}_k|^{\frac{1}{2}}} \tag{26}$$

where $\mathbf{O}$ is the concatenation of the pixel spatial and chromatic characteristics. Since the color and spatial statistics are assumed to be independent, the cross-covariance $\boldsymbol{\Lambda}_{sc}$ goes to zero, and the computation of the above value can proceed in a separable fashion [51].

For a frame of video data, the pixel at (i, j) can be classified by selecting, from the k blobs being tracked, that blob which best predicts the observed pixel:

$$\Gamma_{ij} = \arg\max_k [\Pr(\mathbf{O}_{ij}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)] \tag{27}$$

where Γ_{ij} is the labeling of pixel (i, j). Due to the connected nature of people, it is possible to increase efficiency by growing out from initial position estimates to the outer edge of the figure. This allows the algorithm to only touch the pixels that represent the person and those nearby [51].

3.1.3 Model Update

Once the pixels are labeled by Γ, blob statistics can be re-estimated from the image data. For each class k, the pixels marked as members of the class are used to estimate the new model mean $\boldsymbol{\mu}_k$:

$$\hat{\boldsymbol{\mu}}_k = E[\boldsymbol{O}] \tag{28}$$

and the second-order statistics become the estimate of the model's covariance matrix $\boldsymbol{\Sigma}_k$,

$$\hat{\boldsymbol{\Sigma}}_k = E[(\boldsymbol{O} - \boldsymbol{\mu}_k)(\boldsymbol{O} - \boldsymbol{\mu}_k)^T] \tag{29}$$

3.1.4 A Compact Model

These updated blob statistics represent a low-dimensional, object-based description of the video frame. The position of the blob is specified by the two parameters of the distribution mean vector $\boldsymbol{\mu}_s$: i and j. The spatial extent of each blob is represented by the three free parameters in the covariance matrix $\boldsymbol{\Sigma}_s$. A natural interpretation of these parameters can be obtained by performing the eigenvalue decomposition of $\boldsymbol{\Sigma}_s$:

$$\boldsymbol{\Sigma}_s \begin{bmatrix} | & | \\ L_1 & L_2 \\ | & | \end{bmatrix} = \begin{bmatrix} | & | \\ L_1 & L_2 \\ | & | \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \tag{30}$$

Without loss of generality, $\lambda_1 \geq \lambda_2$, and $\|L_1\| = \|L_2\| = 1$. With those constraints, λ_1 and λ_2 represent the length of the semi-major and semi-minor axes of the iso-probability contour ellipse defined by $\boldsymbol{\Sigma}_s$. The vectors L_1 and L_2 specify the direction of these axes. Since they are perpendicular, they can be specified by a single parameter, say ω, the rotation of the semi-major axis away from the x axis. Thus we can represent the model $\{\boldsymbol{\mu}_s \boldsymbol{\Sigma}_s\}$ with five parameters:

$$\{i, j, \lambda_1, \lambda_2, \omega\}$$

Furthermore, these parameters have the convenient physical interpretation of representing the center, length, width, and orientation of an ellipse in the image plane, as shown in Figure 9.

Since the typical blob is supported by tens to hundreds of pixels, it is possible to robustly estimate these five parameters from the available data. The result is a stable, compact, object-level representation of the image region explained by the blob.

3.1.5 Recovery of a Three Dimensional Model

These 2-D features are the input to the 3-D blob estimation framework used by Azarbayejani and Pentland [4]. This framework relates the 2-D distribution of pixel values to a tracked object's 3-D position and orientation.

Inside the larger recursive framework, this estimation is carried out by an embedded extended Kalman filter. It is the structure from motion estimation

Figure 9: The hand as an iso-probability ellipse.

framework developed by Azarbayejani to estimate 3-D geometry from images. As an extended Kalman filter, it is itself a recursive, nonlinear, probabilistic estimation framework. Estimation of 3-D parameters from calibrated sets of 2-D parameters is computationally very efficient, requiring only a small fraction of computational power as compared to the segmentation algorithms described above[5]. The reader should not be confused by the embedding of one recursive framework inside another: for the larger context this module may be considered an opaque filter.

The same estimation machinery used to recover these 3-D blobs can also be used to quickly and automatically calibrate pairs of cameras from blob observation data[4].

After this estimation process, the 3-D blob has nine parameters:

$$\{x, y, z, l_1, l_2, l_3, w_1, w_2, w_3\}$$

As above, these parameters represent the position of the center of the blob, the length of the fundamental axes defined by an iso-probability contour, and the rotation of these axes away from some reference frame. Figure 10 shows the result of this estimation process.

3.2 Modeling Dynamics

There is 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

Figure 10: The hand as a 3-D blobject.

model consists of the model state, $\mathbf{x}$, and the model parameters, $\mathbf{p}$, where the parameters, $\mathbf{p}$ are the unchanging qualities of the system (the length or mass of a given link, for example).

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: $\mathbf{x}, \dot{\mathbf{x}}$, and the model parameters, $\mathbf{p}$. The state evolves according to Newton's First Law:

$$\ddot{\mathbf{x}} = \mathbf{W} \cdot \mathbf{X} \tag{31}$$

where $\mathbf{X}$ 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.

3.2.1 Hard Constraints

Hard constraints represent absolute limitations imposed on the system. One example is the kinematic constraint of a skeletal joint. The model follows the *virtual work* formulation of Witkin[50]. The Witkin formulation has several advantages over reduced dimensionality solutions such as that described by Featherstone[16]: the constraints can be modified at run-time, and the modularity inherent in the mathematics drastically simplifies the implementation. The one significant disadvantage, which will be addressed below, is computational efficiency.

In a virtual work constraint formulation, all the links in a model have full range of unconstrained motion. Hard kinematic constraints on the system are

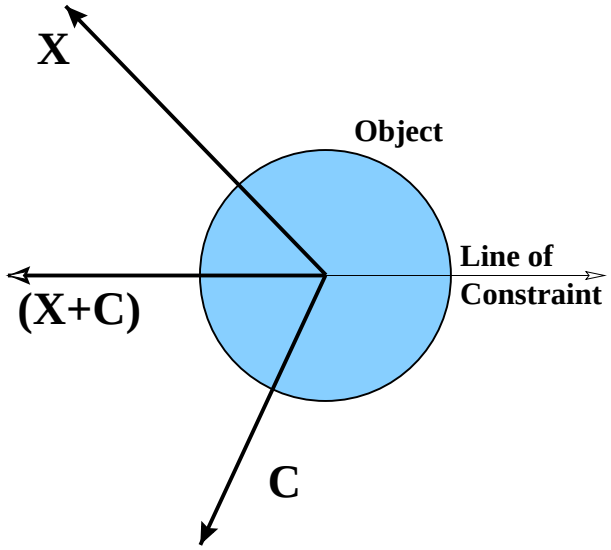

Figure 11: The 2-D object is constrained to move on the indicated line. An external force, **X**, is applied to the object. The constraint force, **C**, keeps the object from accelerating away from the constraint.

enforced by a special set of forces **C**:

$$\ddot{\mathbf{x}} = \mathbf{W} \cdot (\mathbf{X} + \mathbf{C}) \tag{32}$$

The constraints are specified as mathematical relationships between objects that are defined to be zero when the constraint is satisfied. The constraint forces C, are chosen to insure that the constraints stay satisfied. In Figure 11 the constraint would be expressed as the distance between the line and the center of the object.

The constraints are functions of model state and time: $\mathbf{c}(\mathbf{x}, t)$. When a constraint is satisfied then $\mathbf{c} = 0$. If a constraint is to remain satisfied, then the constraint velocity must remain zero, $\dot{\mathbf{c}} = 0$, and the constraint must not accelerate away from a valid state: $\ddot{\mathbf{c}} = 0$ or the system would soon be in an invalid state. The constraint forces from Equation 32 and Figure 11 keep the system from accelerating away from constraint satisfaction. The road to understanding these forces begins by differentiating **c**:

$$\dot{\mathbf{c}} = \frac{\partial}{\partial \mathbf{x}} \mathbf{c}(\mathbf{x}, t) = \frac{\partial \mathbf{c}}{\partial \mathbf{x}} \dot{\mathbf{x}} + \frac{\partial \mathbf{c}}{\partial t} \tag{33}$$

and again:

$$\ddot{\mathbf{c}} = \frac{\partial \mathbf{c}}{\partial \mathbf{x}} \ddot{\mathbf{x}} + \frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{x}} \dot{\mathbf{x}} + \frac{\partial^2 \mathbf{c}}{\partial t^2} \tag{34}$$

Combining with Equation 32 and setting $\ddot{\mathbf{c}} = 0$ yields a relationship between the model state, the applied external forces, and the constraint Jacobian, where

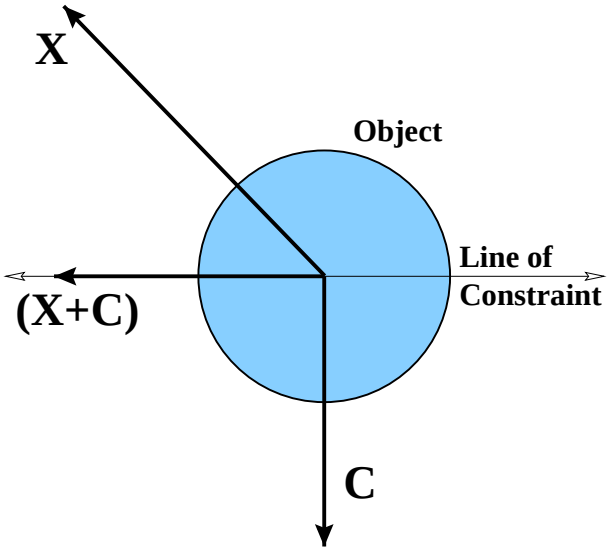

Figure 12: No work is performed if the constraint force lies in the null space compliment of the constraint Jacobian.

the constraint restitution force $\mathbf{C}$ is the only unknown:

$$\frac{\partial \mathbf{c}}{\partial \mathbf{x}}\mathbf{W} \cdot (\mathbf{X} + \mathbf{C}) + \frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{x}}\dot{\mathbf{x}} + \frac{\partial^2 \mathbf{c}}{\partial t^2} = 0 \tag{35}$$

The force in Figure 11 satisfies the relationship in Equation 35, as do many other possible vectors. Equation 35 is an under-determined system since the dimensionality of $\mathbf{c}$ will always be less than the dimensionality of $\mathbf{x}$, or the system would be fully constrained and wouldn't move at all. For example, in Figure 11, $\mathbf{c}$ is the distance between the center of the object and the line, a one dimensional value, while $\mathbf{x}$ is two dimensional, three if the object is allowed rotational freedom in the plane.

One problem with that choice of $\mathbf{C}$ in Figure 11 is that it will add energy to the system. Equation 35 only specifies that the force exactly counteract the component of $\mathbf{X}$ that is in violation of the constraints. Since constraints that add energy to the models will lead to instability, an additional requirement that the force $\mathbf{C}$ do no work on the system is employed, where work would be $\mathbf{C}\dot{\mathbf{x}}$. If $\mathbf{C}$ is always applied perpendicular to any direction of motion allowed by the constraint then this will be satisfied. In the example case of the object on the line, this means that $\mathbf{C}$ must be perpendicular to the line of constraint, as shown in Figure 12.

More generally, the valid displacements for the system are described by the null space of the constraint Jacobian:

$$\frac{\partial \mathbf{c}}{\partial \mathbf{x}} d\mathbf{x} = 0 \tag{36}$$

since valid displacements are required to leave $\mathbf{c} = 0$. The disallowed displacements are ones that do change $\mathbf{c}$:

$$d\mathbf{x} = \boldsymbol{\lambda}\frac{\partial \mathbf{c}}{\partial \mathbf{x}} \tag{37}$$

So, to do no work, the constraint force $\mathbf{C}$ is required to lie in the same subspace:

$$\mathbf{C} = \boldsymbol{\lambda}\frac{\partial \mathbf{c}}{\partial \mathbf{x}} \tag{38}$$

Combining that equation with Equation 35 results in a linear system of equations with only the one unknown, $\boldsymbol{\lambda}$:

$$-\left[\frac{\partial \mathbf{c}}{\partial \mathbf{x}}\mathbf{W}\frac{\partial \mathbf{c}}{\partial \mathbf{x}}^T\right]\boldsymbol{\lambda} = \frac{\partial \mathbf{c}}{\partial \mathbf{x}}\mathbf{W}\mathbf{X} + \frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{x}}\dot{\mathbf{x}} + \frac{\partial^2 \mathbf{c}}{\partial t^2} \tag{39}$$

This equation can be rewritten to emphasize its linear nature. $\mathbf{J}$ is the constraint Jacobian, $\boldsymbol{\rho}$ is a known constant vector, and $\boldsymbol{\lambda}$ is the vector of unknown Lagrange multipliers:

$$-\mathbf{J}\mathbf{W}\mathbf{J}^T\boldsymbol{\lambda} = \boldsymbol{\rho} \tag{40}$$

To obtain $\mathbf{C}$, $\boldsymbol{\lambda}$ is substituted back into Equation 38. Many fast, stable methods exist for solving equations of this form.

All the components of Equation 39 that relate directly to the constraints are linearizations, so they must be recomputed at each integration step. This provides the opportunity to create and delete constraints at run-time simply by modifying the calculation of the constraint Jacobian.

Modularization Constraints are written as mathematical relationships between points in space. Often these points are located at some arbitrary location in the local coordinate space of some object. Implementing constraints to understand each type of object, possibly having different internal representations for state, would make the constraints unnecessarily complex. Witkin suggests inserting an abstraction layer between objects and constraints, called connectors[50]. Thus $\mathbf{c}$ for a constraint between two objects becomes:

$$\mathbf{c}(\mathbf{x}) = f(\mathbf{a}(\mathbf{x}_1), \mathbf{b}(\mathbf{x}_2)) \tag{41}$$

The constraint Jacobian can then be decomposed by the chain rule:

$$\frac{\partial \mathbf{c}}{\partial \mathbf{x}} = \frac{\partial \mathbf{c}}{\partial \mathbf{a}}\frac{\partial \mathbf{a}}{\partial \mathbf{x}_1} + \frac{\partial \mathbf{c}}{\partial \mathbf{b}}\frac{\partial \mathbf{b}}{\partial \mathbf{x}_2} \tag{42}$$

The constraint module can then compute $\frac{\partial \mathbf{c}}{\partial \mathbf{a}}$ and $\frac{\partial \mathbf{c}}{\partial \mathbf{b}}$ without regard to the underlying implementation while the connectors are responsible for calculating $\frac{\partial \mathbf{a}}{\partial \mathbf{x}}$. The constraint velocity Jacobian can be computed in the same way;

$$\frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{x}} = \frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{a}}\frac{\partial \mathbf{a}}{\partial \mathbf{x}_1} + \frac{\partial \dot{\mathbf{c}}}{\partial \dot{\mathbf{a}}}\frac{\partial \dot{\mathbf{a}}}{\partial \mathbf{x}_1} + \frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{b}}\frac{\partial \mathbf{b}}{\partial \mathbf{x}_2} + \frac{\partial \dot{\mathbf{c}}}{\partial \dot{\mathbf{b}}}\frac{\partial \dot{\mathbf{b}}}{\partial \mathbf{x}_2} \tag{43}$$

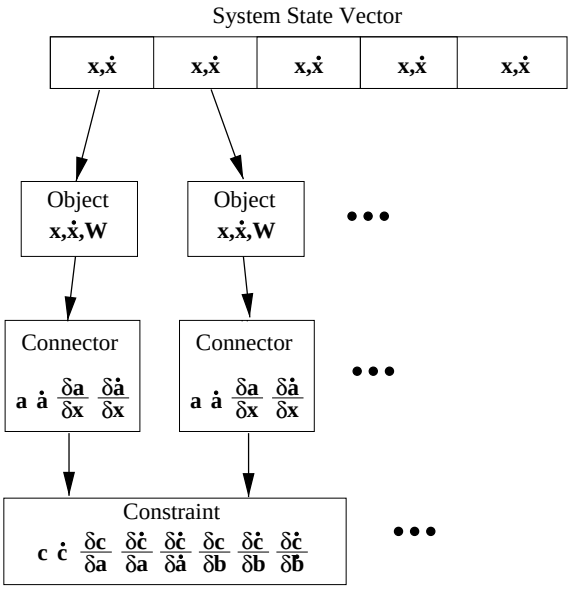

Figure 13: The constraint system is made up of software modules that cooperate to construct Equation 39. Connectors are an abstraction that allows the Constraints to be more general, and hence more reusable.

Figure 13 shows how this information moves through the system. Each block represents a different software module. This abstraction is very powerful: in the system the same constraint code implements pin joints in 2-D models and ball joints in 3-D models. Because constraints involving rotational motion are somewhat more complex, the system differentiates between connectors without orientation, called Points, and connectors with orientation, called Handles.

Multiple Objects and Constraints Systems with only a single constraint are rather limiting. Multiple objects and constraints fit easily into the framework. For multiple objects, the state vector $\mathbf{x}$ becomes the concatenation of all the individual object state vectors. So in a 3-D model where every object has 6 degrees of freedom, with 5 objects the state vector would have dimensionality 30.

The mass matrix is similarly the concatenation of the individual mass matrices. Assuming static geometry for each object, the individual mass matrix is constant in the object local coordinate system. This mass matrix is transformed to global coordinates and added as a block to the global mass matrix. Since the global mass matrix is block diagonal, the inverse mass matrix is simply the concatenation of the individually inverted mass matrices, and so doesn't take an inordinate amount of time to compute.

Objects are enumerated with the order that they contribute to the global

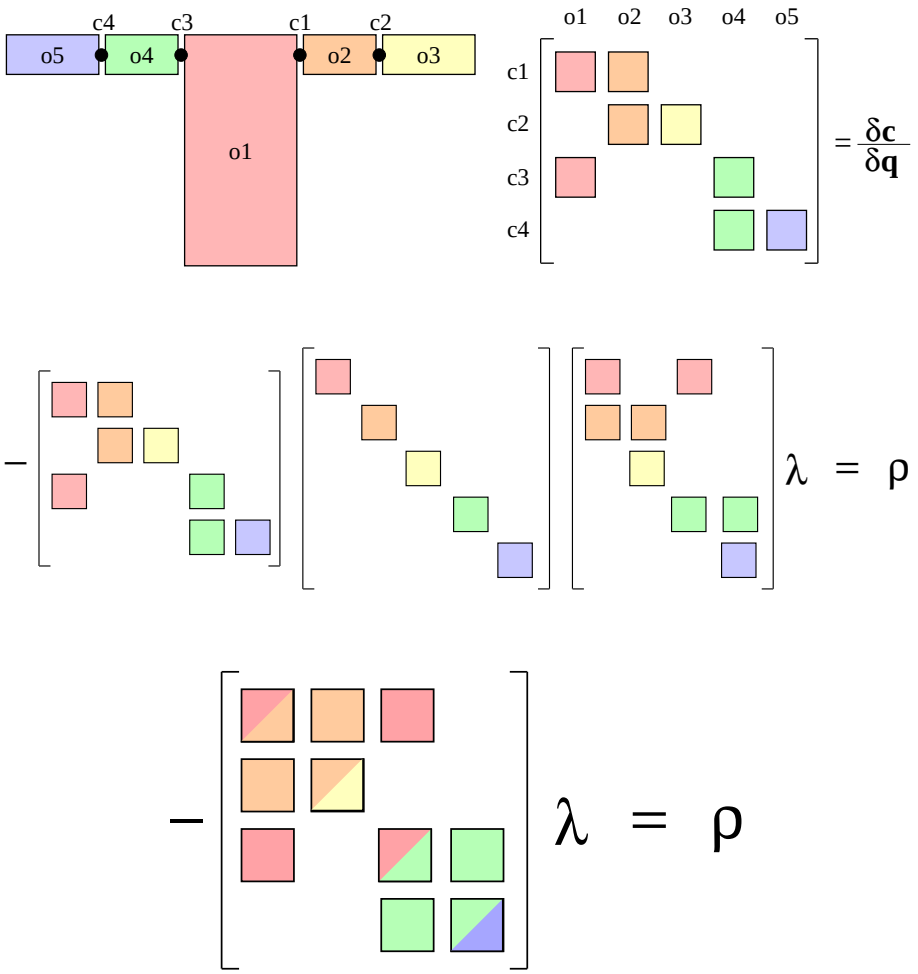

Figure 14: **Top:** the individual constraint Jacobians each contribute one block per object that they affect to the global constraint Jacobian. **Middle:** each object also contributes to the block-diagonal inverse mass matrix from Equation 40. **Bottom:** Sparsely connected systems result in a block-sparse linear system.

state vector. Constraints are similarly enumerated. A constraint between two objects contributes two blocks to the constraint Jacobian. The constraints appear on the row according to the constraint's enumeration and the columns associated with the constrained objects. The structure of the constraint Jacobian is illustrated in Figure 14 for a model of the upper body with five links: torso, left upper arm, left lower arm, right upper arm, and right lower arm. The other values in Equation 39 are constructed in a similar fashion.

The global inverse mass matrix is block diagonal and the global constraint Jacobian is block sparse. Both are large. Solving Equation 39 for $\boldsymbol{\lambda}$ requires sparse matrix methods to be accomplished efficiently. Sparse matrix methods were used to construct $\mathbf{JWJ}^T$. An implementation of Linear Biconjugate Gradient Descent for sparse matrices was used to solve the resulting linear system. The algorithms were taken from Numerical Recipes[39]. These improvements made the constraint system tractable on contemporary hardware. The rest of the matrix manipulations are handled with a basic C++ matrix library.

Discretization Error The constraints of Equation 39 are only true instantaneously. When the equations are solved at discrete time steps then errors are introduced and the system drifts away from the manifold of valid states. A restoring force is used to keep the system from accumulating errors over time:

$$\ddot{\mathbf{x}} = \mathbf{W} \cdot (\mathbf{X} + \mathbf{C} + \mathbf{F}) \tag{44}$$

Where $\mathbf{F}$ is determined by the relationship:

$$F = \alpha \mathbf{c} \frac{\partial \mathbf{c}}{\partial \mathbf{x}} + \beta \dot{\mathbf{c}} \frac{\partial \mathbf{c}}{\partial \mathbf{x}} \tag{45}$$

This applies a restoring force in the constrained direction that brings the system back toward the nearest valid state and a damping force that reduces illegal velocity. The parameters α and β are fixed. In practice the selection of these parameters has very little impact on model stability since deviations from constraints remain small. A typical value for α is $1000\frac{N}{m}$ and a typical value for β is $4\frac{Ns}{m}$.

Distributed Integration Once the global forces are projected back into the allowable subspace and corrected for discretization error, all further computation is partitioned among the individual objects. This avoids computing the very large global version of Equation 32. This is possible since the inverse mass matrix $\mathbf{W}$ is block diagonal, so once the global value for $\mathbf{C}$ is determined, Equation 32 breaks down into a set of independent systems. This distributed force application and integration also provides the opportunity for objects to transform the applied forces to the local frame and to deal with forces and torques separately. This simplifies the implementation of the dynamics subsystem significantly, since each link is treated as a free six degree of freedom body.

3.2.2 Soft Constraints

Some constraints are probabilistic in nature. Noisy image measurements are a constraint of this sort, they influence the dynamic model but do not impose hard constraints on its behavior. As a result, the absolute constraint satisfaction described in the previous section is not appropriate.

Soft constraints are more appropriately expressed as a potential field acting on the dynamic system. The addition of a potential field function to model a probability density function pushes the model toward the most likely value. In general a soft constraint might be any function:

$$\mathbf{X}_{\text{soft}} = f_{\text{soft}}(\mathbf{S}, \mathbf{x}, \dot{\mathbf{x}}, \mathbf{p}) \tag{46}$$

where $\mathbf{S}$ is some parameterization over the family of potential fields specified by $f(\cdot)$.

The simplest function is the constant potential. Gravity is well-modeled by a constant field over the scales of the model. So the potential field is simply:

$$\mathbf{X}_{\text{soft}} = mg \tag{47}$$

where g is acceleration due to gravity, and m is the mass of the link affected by $\mathbf{X}_{\text{soft}}$.

A soft constraint that attracts a body part to a specific location is somewhat more complex:

$$\mathbf{X}_{\text{soft}} = k(\mathbf{x}_0 - \mathbf{x}) \tag{48}$$

where $\mathbf{x}_0$ is the desired position and k is a constant multiplier that affect the "softness" of the constraint. Care must be taken when choosing k to avoid introducing instabilities into the model. Values of k that are too large start to turn the soft constraint into something more like a hard constraint. In this case the constraint would be better modeled by the techniques described above.

It is also possible to construct anisotropic constraints

$$\mathbf{X}_{\text{soft}} = \frac{(\mathbf{x} - \mathbf{x}_0)}{\|(\mathbf{x} - \mathbf{x}_0)\|}(\mathbf{x} - \mathbf{x}_0)K^{-1}(\mathbf{x} - \mathbf{x}_0) \tag{49}$$

where K is a shaping matrix that determines the weighting of various directions. This allows soft constraints to have stronger influence in a particular direction. This is useful for modelling the influence of the blob observations discussed above, or any other regular, non-isotropic force field.

Note that functions may be arbitrarily complex. A good example is a controller of the form described in Section 3.4. Despite their complexity, the dynamics engine may represent them as a time-varying potential field. The forces applied by the controller simply become another force affecting the dynamic evolution of the model. The neuroscience literature supports this model[43].

3.2.3 Observation Influence

The 3-D observations described in Section 3.1 supply constraints on the underlying 3-D human model. Due to their statistical nature, observations are easily

modeled as soft constraints. Observations are integrated into the dynamic evolution of the system by describing them with potential fields, as discussed in Section 3.2.2. These potential fields apply forces to the body model, causing it evolve over time toward the observations. The strength of these fields is related to the Kalman gain in a classic Kalman filter.

3.3 The Inverse Observation Model

In the open-loop system, the vision system uses a Maximum Likelihood framework to label individual pixels in the scene (Equation 27). To close the loop, we need to incorporate information from the 3-D model. This means generating 2-D statistical models from the 3-D body model that can be utilized by the vision system to improve its decisions.

The current state of the model $(\mathbf{x}_t, \dot{\mathbf{x}}_t)$ specifies the best estimate of the configuration of the body in model space given past observations. The first step in predicting future observations is to propagate the state forward according to the dynamic constraints described above. The external forces acting on the system can be assumed to be constant for the period of forward prediction (33ms in the case of video rate observations), or can be predicted forward in time by behavior models as described below in Section 3.4.

Once a best estimate of the future configuration of the system, $\mathbf{x}_{t+\Delta}$, has been computed, the next step is to generate a set of hypothetical 3-D observations that we would expect to see given that configuration. This involves generating distributions that represent 3-D ellipsoids that fit the observable portions of the model links. These distributions are transformed into the camera reference frame as described in Section 3.1.5. In this frame they are described as in Section 3.1 by either their second order statistics:

$$\{\boldsymbol{\mu}_k^\circ, \boldsymbol{\Sigma}_k^\circ\}$$

or, by their free parameters:

$$\{x, y, z, l_1, l_2, l_3, w_1, w_2, w_3\}$$

The process of identifying the observable portions of these links is discussed in Section 3.2.3.

To be used as a prior for the classification decision in Equation 27, these 3-D distributions must be rendered into 2-D image coordinates using perspective projection. These projected distribution will be described by their second order statistics:

$$\{\boldsymbol{\mu}_k^\star, \boldsymbol{\Sigma}_k^\star\}$$

or alternatively by the free parameters:

$$\{i, j, \lambda_1, \lambda_2, \omega\}$$

The computation of $\boldsymbol{\mu}_k^\star$ from $\boldsymbol{\mu}_k^\circ$ is a straightforward application of the forward projective camera model. The parameters x, y, z map into the parameters

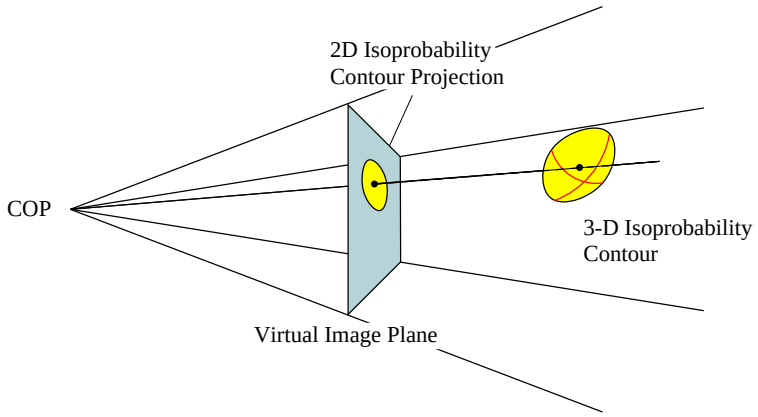

Figure 15: Projecting a 3-D blob into a 2-D image plane.

i, j by perspective projection:

$$\boldsymbol{\mu}_k^\star = \begin{bmatrix} i \\ j \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \frac{1}{1+z} \tag{50}$$

The true perspective projection of a 3-D Gaussian distribution over $\Re^3$ is not a Gaussian distribution over the image coordinates $\Re^2$. It is necessary to employ an approximation to perspective projection that will yield a Gaussian distribution in image coordinates to obtain a value for $\boldsymbol{\Sigma}_k^\star$

Orthographic projection of a Gaussian distribution does result in a Gaussian distribution. This process involves integrating over the Gaussian in the direction of projection. Orthographic projection of the 3-D prior onto a XY plane passing through the mean is thus equivalent to taking the marginal of a zero-mean Gaussian distribution:

$$\mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \sigma_x & \lambda_{xy} \\ \lambda_{yx} & \sigma_y \end{bmatrix}\right) = \int_{-\infty}^{\infty} \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \sigma_x & \lambda_{xy} & \lambda_{xz} \\ \lambda_{yx} & \sigma_y & \lambda_{yz} \\ \lambda_{zx} & \lambda_{zy} & \sigma_z \end{bmatrix}\right) \partial z \tag{51}$$

Orthographic projection does not account for the scaling effect of perspective projection, so simply using orthographic projection would result in priors with significantly exaggerated covariances. A solution is to use the scaled orthographic approximation to perspective projection. Scaled orthographic projection uses perspective projection to map an intermediate 2-D orthographic projection into the virtual image plane. Since the plane of orthographic projection is parallel to the virtual image plane, this operation is equivalent to a scale. Scaling a Gaussian distribution retains the Gaussian nature, so we have the approximation we need. As illustrated in Figure 16, by placing the plane of orthographic projection at z, we can compute the 2-D blob covariance prior,

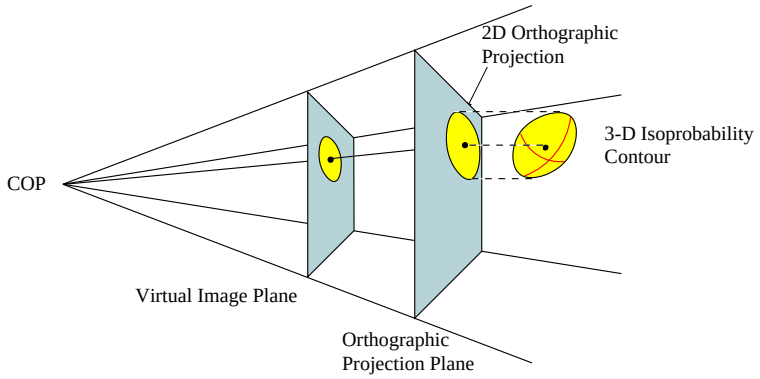

Figure 16: Scaled-Orthographic projection approximation for $\mathbf{\Sigma}_k^\star$

$\mathbf{\Sigma}_k^\star$, from the 3-D covariance $\mathbf{\Sigma}_k^\circ$:

$$\mathbf{\Sigma}_k^\star = \begin{bmatrix} \sigma_i & \lambda_{ij} \\ \lambda_{ji} & \sigma_j \end{bmatrix} = \begin{bmatrix} \frac{1}{1+z} & 0 \\ 0 & \frac{1}{1+z} \end{bmatrix} \begin{bmatrix} \sigma_x & \lambda_{xy} \\ \lambda_{yx} & \sigma_y \end{bmatrix} \begin{bmatrix} \frac{1}{1+z} & 0 \\ 0 & \frac{1}{1+z} \end{bmatrix} \tag{52}$$

The result of this process is a prior distribution on image observations in the next frame:

$$\Pr(\mathbf{O}_{ij}|\boldsymbol{\mu}_k^\star, \mathbf{\Sigma}_k^\star) \tag{53}$$

Integrating this information into the 2-D statistical decision framework of Equation 27 results in a Maximum *A Posteriori* decision rule for pixel classification:

$$\Gamma_{ij} = \arg\max_k \left[\Pr(\mathbf{O}_{ij}|\boldsymbol{\mu}_k, \mathbf{\Sigma}_k)^\alpha \cdot \Pr(\mathbf{O}_{ij}|\boldsymbol{\mu}_k^\star, \mathbf{\Sigma}_k^\star)^{1-\alpha}\right] \tag{54}$$

where α is the weighting parameter that indicates the importance of the prior information:

$$0 \leq \alpha \leq 1 \tag{55}$$

3.4 A Model for Control

Observations of the human body reveal an interplay between the passive evolution of a physical system (the human body) and the influences of an active, complex controller (the nervous system). Section 3.2 explains how, with a bit of work, it is possible to model the physical aspects of the system. However, it is *very* difficult to explicitly model the human nervous and muscular systems, so the approach of using observed data to estimate probability distributions over control space is very appealing.

3.4.1 A Model for Control

Kalman filtering includes the concept of an *innovations process*. This is the difference between the actual observation and the predicted observation trans-

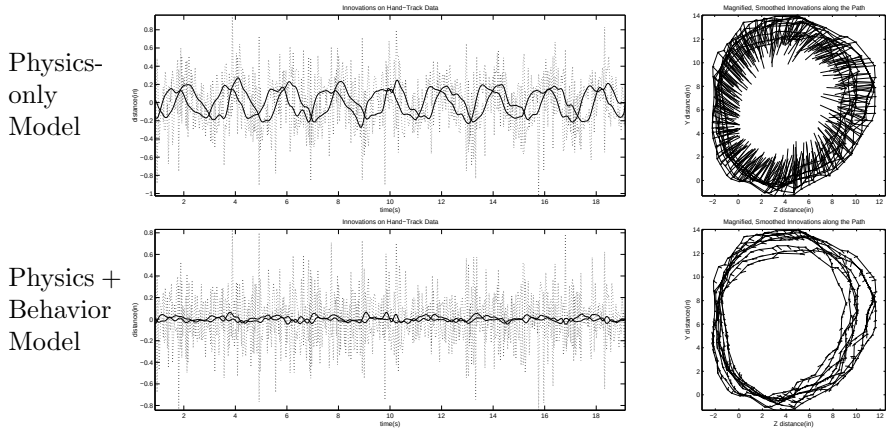

Figure 17: Modeling tracking data of circular hand motion. Passive physics alone leaves significant structure in the innovations process. **Top Left:** Smoothing the innovations reveals unexplained structure. **Top Right:** Plotting the Innovations along the path makes the purposeful aspect of the action clear. **Bottom:** In this example, using a learned control model to improve predictions leaves only white process noise in the innovations process. The smoothed innovations stay near zero.

formed by the Kalman gain:

$$\boldsymbol{\nu}_t = \mathbf{K}_t(\mathbf{y}_t - \mathbf{H}_t\boldsymbol{\Phi}_t\hat{\mathbf{x}}_{t-1}) \tag{56}$$

The innovations process $\boldsymbol{\nu}$ is the sequence of information in the observations that was not adequately predicted by the model. If we have a sufficient model of the observed dynamic process, and white, zero-mean Gaussian noise is added to the system, either in observation or in the real dynamic system itself, then the innovations process will be white. Inadequate models will cause correlations in the innovations process.

Since purposeful human motion is not well modeled by passive physics, we should expect significant structure in the innovations process.

A simple example is helpful for illustrating this idea. If we track the hand moving in a circular motion, then we have a sequence of observations of hand position. This sequence is the result of a physical thing being measured by a noisy observation process. For this simple example we making the assumption that the hand moves according to a linear, constant velocity dynamic model. Given that assumption, it is possible to estimate the true state of the hand, and predict future states and observations. If this model is sufficient, then the errors in the predictions should be solely due to the noise in the system.

The upper plots in Figure 17 show that model is not sufficient. Smoothing $\boldsymbol{\nu}$ reveals this significant structure (top left). Plotting the innovations along the path of observations makes the relationship between the observations and

the innovations clear: there is some un-modeled process acting to keep the hand moving in a circular motion (top right). This un-modeled process is the purposeful control signal that being applied to the hand by the muscles.

In this example, there is one active, cyclo-stationary control behavior, and its relationship to the state of the physical system is straightforward. There is a one-to-one mapping between the state and the phase offset into the cyclic control, and a one-to-one mapping between the offset and the control to be applied. If we use the smoothed innovations as our model and assume a linear control model of identity, then the linear prediction becomes:

$$\hat{\mathbf{x}}_t = \mathbf{\Phi}_t \hat{\mathbf{x}}_{t-1} + \mathbf{I}\mathbf{u}_{t-1} \tag{57}$$

where $\mathbf{u}_{t-1}$ is the control signal applied to the system. The lower plots in Figure 17 show the result of modeling the hand motion with a model of passive physics and a model of the active control. The smoothed innovations are basically zero: there is no part of the signal that deviates from our model except for the observation noise.

In this simple, linear example the system state, and thus the innovations, are represented the same coordinate system as the observations. With more complex dynamic and observations models, such as described in Section 3.2, they could be represented in any arbitrary system, including spaces related to observation space in non-linear ways, for example as joint angles.

The next section examines a more powerful form of model for control.

3.4.2 Multiple Behavior Models

Human behavior, in all but the simplest tasks, is not as simple as a single dynamic model. The next most complex model of human behavior is to have *several* alternative models of the person's dynamics, one for each class of response. Then at each instant we can make observations of the person's state, decide which model applies, and then use that model for estimation. This is known as the *multiple model* or *generalized likelihood* approach, and produces a generalized maximum likelihood estimate of the current and future values of the state variables [49]. Moreover, the cost of the Kalman filter calculations is sufficiently small to make the approach quite practical.

Intuitively, this solution breaks the person's overall behavior down into several "prototypical" behaviors. For instance, we might have dynamic models corresponding to a relaxed state, a very stiff state, and so forth. We then classify the behavior by determining which model best fits the observations. This is similar to the multiple model approach of Friedmann, and Isard[18, 24].

Since the innovations process is the part of the observation data that is unexplained by the dynamic model, the behavior model that explains the largest portion of the observations is, of course, the model most likely to be correct. Thus, at each time step, we calculate the probability $Pr^{(i)}$ of the m-dimensional observations $\mathbf{Y}_k$ given the i^{th} model and choose the model with the largest probability. This model is then used to estimate the current value of the state

variables, to predict their future values, and to choose among alternative responses.

3.4.3 Hidden Markov Models of Control

Since human motion evolves over time, in a complex way, it is advantageous to explicitly model temporal dependence and internal states in the control process. A Hidden Markov Model (HMM) is one way to do this, and has been shown to perform quite well recognizing human motion[46].

The probability that the model is in a certain state, S_j, given a sequence of observations, $\mathbf{O}_1, \mathbf{O}_2, \ldots, \mathbf{O}_N$, is defined recursively. For two observations, the density associated with the state after the second observation, $\mathbf{q}_2$, being S_j is:

$$\Pr(\mathbf{O}_1, \mathbf{O}_2, \mathbf{q}_2 = S_j) = \left[\sum_{i=1}^{N} \pi_i b_i(\mathbf{O}_1) \mathbf{a}_{ij} \right] b_j(\mathbf{O}_2) \tag{58}$$

where π_i is the prior probability of being in state i, and $b_i(\mathbf{O})$ is the probability of making the observation $\mathbf{O}$ while in state i. This is the Forward algorithm for HMM models.

Estimation of the control signal proceeds by identifying the most likely state given the current observation and the last state, and then using the observation density of that state as described above. If the models are trained relative to a passive-physics model, then likely it will be necessary to run a passive-physics tracker to supply the innovations that will be used by the models to select the control paradigm for a second tracker. We restrict the observation densities to be either a Gaussian or a mixture of Gaussians. For behaviors that are labeled there are well understood techniques for estimating the parameters of the HMM from data[40].

3.4.4 Behavior Alphabet Auto-Selection

Classic HMM techniques require the training data to be labeled prior to parameter estimation. Since we don't necessarily know how to choose a gesture alphabet *a priori*, we cannot perform this pre-segmentation. We would prefer to automatically discover the optimal alphabet for gestures from gesture data. The COGNO architecture performs this automatic clustering[12].

Unfortunately, the phrase "optimal" is ill-defined for this task. In the absence of a task to evaluate the performance of the model, there is an arbitrary trade-off between model complexity and generalization of the model to other data sets[48]. By choosing a task, such as discriminating styles of motion, we gain a well-defined metric for performance.

One of our goals is to observe a user who is interacting with a system and be able to automatically find patterns in their behavior. Interesting questions include:

- Is this (a)typical behavior for the user?

- Is this (a)typical behavior for anyone?
- When is the user transitioning from one behavior/strategy to another behavior/strategy?
- Can we do filtering or prediction using models of the user's behavior?

We must find the behavior alphabets that pick out the salient movements relevant to the above questions. There probably will not be one canonical alphabet for all tasks but rather many alphabets each suited to a group of tasks. Therefore we need an algorithm for automatically generating and selecting effective behavior alphabets. The goal of finding an alphabet that is suitable for a machine learning task can be mapped to the concept of feature selection.

The examples presented in Section 4 employed the COGNO Algorithm[12] to perform unsupervised clustering of the passive-physics innovations sequences. Unsupervised clustering of temporal sequences generated by human behavior is a very active topic in the literature[45, 1, 32, 38].

3.5 Summary

This section presents a framework for human motion understanding, defined as estimation of the physical state of the body combined with interpretation of that part of the motion that cannot be predicted by passive physics alone. The behavior system operates in conjunction with a real-time, fully-dynamic, 3-D person tracking system that provides a mathematically concise formulation for incorporating a wide variety of physical constraints and probabilistic influences. The framework takes the form of a non-linear recursive filter that enables pixel-level processes to take advantage of the contextual knowledge encoded in the higher-level models.

The intimate integration of the behavior system and the dynamic model also provides the opportunity for a richer sort of motion understanding. The innovations are one step closer to the original intent, so the statistical models don't have to disentangle the message from the means of expression.

Some of the benefits of this approach including increase in 3-D tracking accuracy, insensitivity to temporary occlusion, and the ability to handle multiple people will be demonstrated in the next section.

4 Results

This section will provide data to illustrate the benefits of the DYNA framework. The first part will report on the state of the model within DYNA and the quantitative effects of tracking improvements. The rest will detail qualitative improvements in human-computer interface performance in the context of several benchmark applications.

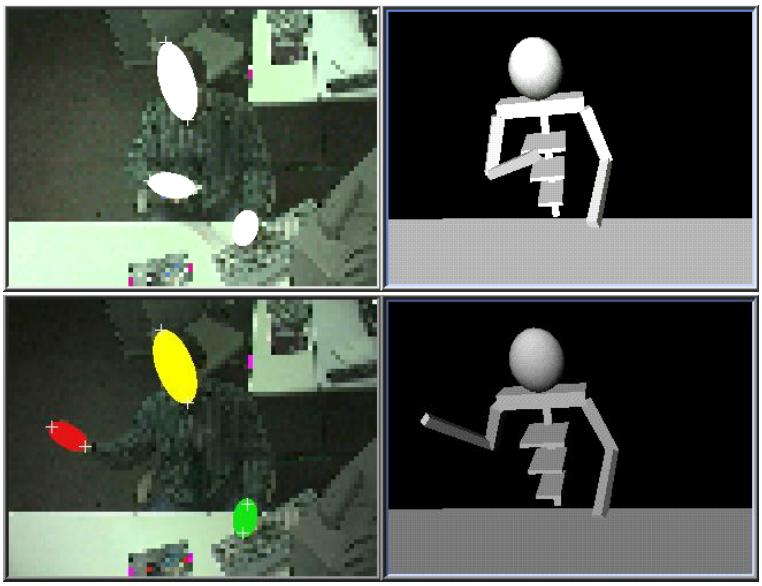

Figure 18: **Left:** video and 2-D blobs from one camera in the stereo pair. **Right:** corresponding configurations of the dynamic model.

4.1 Tracking Results

The dynamic skeleton model currently includes the upper body and arms. The full dynamic system loop, including forward integration and constraint satisfaction, iterates on a 500MHz Alpha 21264 at 600Hz. Observations come in from the vision system at video rate, 30Hz, so this is sufficiently fast for real-time operation. Figure 18 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, the kinematic constraints of the skeleton, and and the behavior (control) model.

The model also rejects noise that is inconsistent with the dynamic model. This process isn't equivalent to a simple isometric smoothing, since the mass matrix of the body is anisotropic and time-varying. When combined with an active control model, tracking error can be further reduced through the elimination of overshoot and other effects. Table 19 compares noise in the physics+behavior tracker with the physics-only tracker noise. It can be seen that there is a significant increase in performance.

The plot in Figure 20 shows the observed and predicted X position of the hand and the corresponding innovation trace before, during and after the motion is altered by a constraint that is modeled by the system: arm kinematics. When the arm reaches full-extension, the motion is arrested. The system is able to predict this even and the near-zero innovations after the event reflect this. Non-

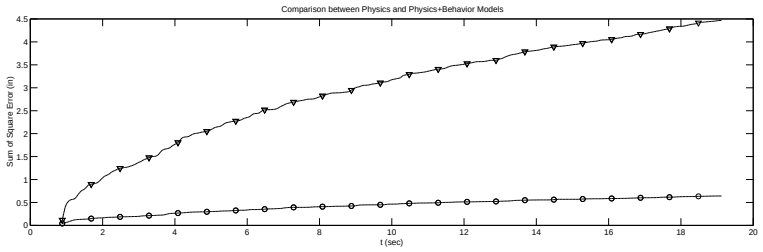

Figure 19: Sum Square Error of a Physics-only tracker (triangles) vs. error from a Physics+Behavior Tracker

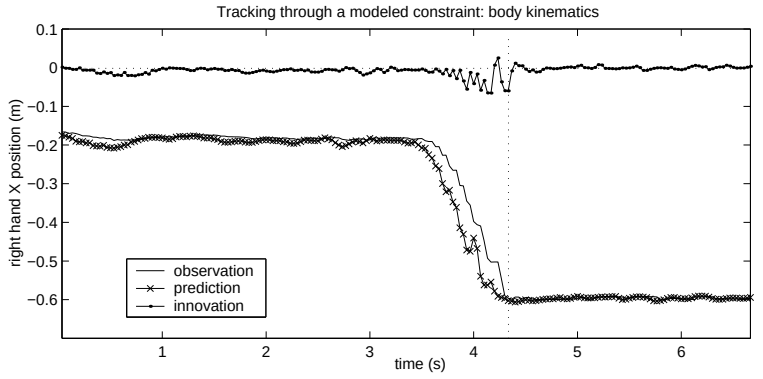

Figure 20: Observed and predicted X position of the hand and the corresponding innovation trace before, during and after expression of a modeled constraint: arm kinematics.

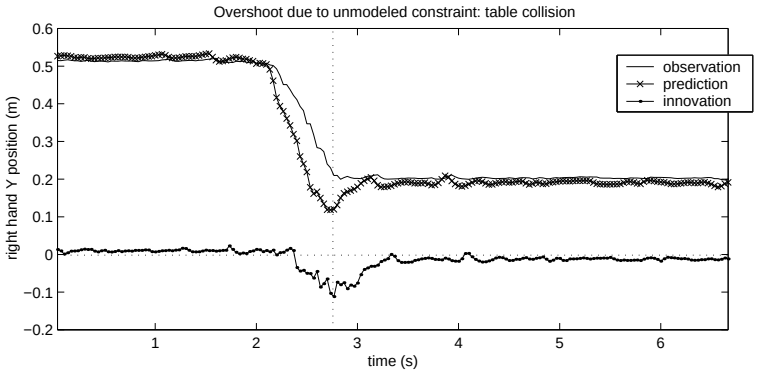

Figure 21: Observed and predicted Y position of the hand and the corresponding innovation trace before, during and after expression of a un-modeled constraint: collision with a table.

zero innovations before the event represent the controlled acceleration of the arm in the negative X direction. Compare to the case of a collision between a hand and the table illustrated in Figure 21. The table is not included in the system's model, so the collision goes unpredicted. This results in overshoot, and a corresponding signal in the innovations process after the event.

Figure 22 illustrates one of the most significant advantages to tracking of feedback from higher-level models to the low-level vision system. The illustrated sequence is difficult to track due to the presence of periodic, binocular, flesh-flesh occlusions. That is, one hand is occluded by the other from both camera viewpoints in a periodic fashion: in this example at approximately 1Hz. The binocular nature of the occlusion events doesn't allow for view selection to aid tracking: there is no unambiguous viewpoint available to the system. Flesh-flesh occlusions are particularly difficult for tracking systems since it's easier to get distracted by an object with similar appearance (like another hand) than it is to be distracted by an object with a very different appearance (like a green shirt sleeve). The periodic nature of the occlusions means that the system only has a limited number of unambiguous observations to gather data before another occlusion again disrupts tracker stability.

Without feedback, the 2-D tracker fails if there is even partial self-occlusion, or occlusion of an object with similar appearance (such as another person), from a single camera's perspective. In the even more demanding situation of periodic, binocular, flesh-flesh occlusions, the tracker fails horribly. The middle pair of plots in Figure 22 show the results. The plots from a cross-eyed stereo pair. The low-level trackers fail at every occlusion causing the instantaneous jumps in apparent hand position reported by the system. Time is along the X axis, from left to right. The other two axes represent Y and Z position of the two hands. The circular motion was performed in the Y-Z plane, so X motion was negligible. It is not shown in the plot.

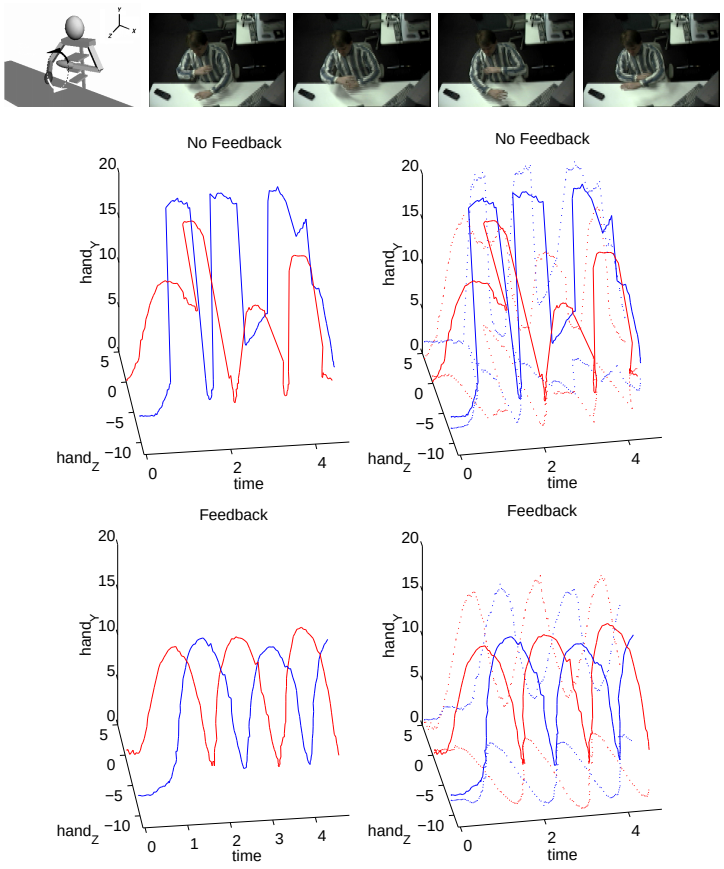

Figure 22: Tracking performance on a sequence with significant occlusion. **Top:** A diagram of the sequence and a single camera's view of **Middle:** A graph of tracking results without feedback (cross-eyed stereo pair). **Bottom:** Correct tracking when feedback is enabled (cross-eyed stereo pair).

The situation with feedback, as illustrated in the lower pair of plots in Figure 22, is much better. Predictions from the dynamic model are used to resolve ambiguity during 2-D tracking. The trackers survive all the occlusions and the 3-D estimates of hand position reveal a clean helix through time (left to right), forming rough circles in the Y-Z plane. With models of behavior, longer occlusions could be tolerated.

4.2 Applications

Section 4.1 provided quantitative measures of improvement in tracking performance. This section will demonstrate improvements in human-computer interaction by providing case studies of several complete systems that use the

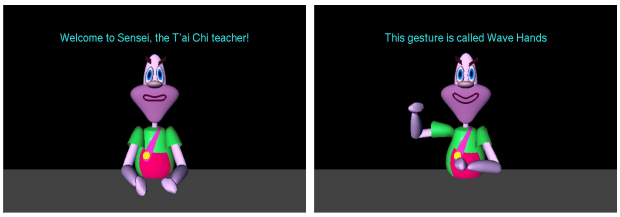

Figure 23: The T'ai Chi sensei shows gives verbal instruction and uses it's virtual body to show the student the T'ai Chi moves.

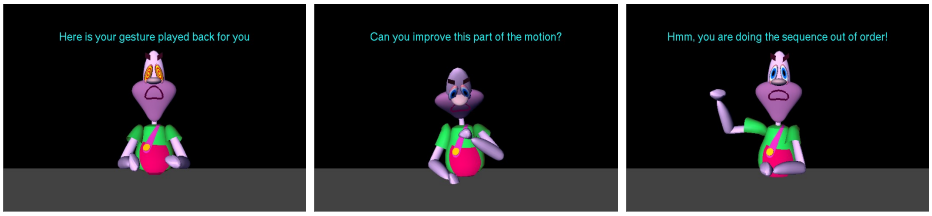

Figure 24: Visual and Auditory cues are used to give the student feedback. The sensei mimics the student motions and indicates problems to be worked on.

perceptual machinery described in Section 3.

The three cases are the T'ai Chi instructional system, the Whack-a-Wuggle virtual manipulation game, and the strategy game Netrek.

4.2.1 T'ai Chi

The T'ai Chi sensei is an example of an application that is significantly enhanced by the recursive framework for motion understanding simply by benefiting from the improved tracking stability. The sensei is an interactive instructional system that teaches the human a selection of upper-body T'ai Chi gestures[7].

The sensei is embodied in a virtual character. That character is used to demonstrate gestures, to provide instant feedback by mirroring the student actions, and to replay the student motions with annotation. Figure 4.2.1 shows some frames from the interaction: the sensei welcoming the student on the left, and demonstrating one of the gestures on the right. The interaction is accompanied by an audio track that introduces the interaction verbally and marks salient events with musical cues.

There are several kinds of feedback that the sensei can provide to the student. The first is the instant gratification associated with seeing the sensei mirror their motions. This allows the student to know that the sensei is attending to their motions and gives immediate feedback regarding their perceived posture relative to the ideal gestures they were just shown. When the sensei decides that feedback is appropriate this mirroring stops: indicating to the student that the interaction paradigm has changed. In this feedback mode the sensei can

either critique individual gestures or remind the user of the global order of the sequence. The left and center images in Figure 24 show an example of a critique of a specific gesture. Visual and musical cues indicate the temporal location of the error and the sensei's gaze direction indicates the errant hand. The right image in Figure 24 is an example of feedback regarding the overall structure of the T'ai Chi sequence.

There are several technologies at work making these interactions possible. The mirroring is accomplished simply by piping the tracking data through the inverse-kinematics engine inside the sensei's body. This is technically simple, but it is imperative that it be robust. It is a meaningful event when the sensei stops mirroring: it signals to the student that they should stop and prepare to receive feedback. Tracking failures can cause the sensei to pause for an indeterminate length of time. Even worse, tracking failures can cause the sensei to spontaneously contort into one of many unpleasant configurations. Either event will obviously detract from the student's learning experience.

The technology behind the identification and interpretation of T'ai Chi gestures is somewhat more complex. To summarize: the sensei learns T'ai Chi gestures by watching a human perform the motions. The system builds Hidden Markov Model (HMM) of each gesture. The HMMs are comprised of a sequence of states with Markov temporal dynamics and Gaussian output probabilities. In effect they are capturing a mean path through parameter space that represents the gesture, and a covariance that specifies an envelope around the mean path that represents the observed variability. The gestures are recognized in the usual way [10]. Once a gesture is recognized, the lattice is examined to find the point at which the observed gesture differ the most from the learned ideal, weighted by the allowable variation [7]. Tracking errors can have a large impact on this process. If a critical chunk of a gesture is missed due to tracking error then it may be unrecognizable, or worse, the system may attempt to correct illusory motion errors and confuse the student.

The individual T'ai Chi gestures are relatively complex. Each gesture takes several second to perform. Critiques often involve playback of observed and ideal gestures, and as a result a single critique may last several tens of seconds. The result is that the frequency of interaction is relatively low. Tracking errors that result in misclassified gestures or illusory motion errors can thus lead to a significant break in the experience. Failures thus lead to frustration. Frustration is antithetical to the underlying goal of T'ai Chi: relaxation and focus.

The bottom (left) plot of Figure 25 shows a flawless execution of a five gesture sequence as tracked by a bottom-up 3-D blob tracker. The individual gestures are hand-labeled at the top of the plot. The plots show, from top to bottom, the X, Y and Z position of the left hand, right hand, and head. Y is positive up. X is positive to the student's right. Z is positive away from the user. The following discussion will focus on two salient features. The first feature is the initial bump in the Y axis of both hands. This corresponds to the up and down motion of "Opening Form". The second feature is the double bump in the right hand X position at the end of the sequence. This corresponds to the right hand motion in "Single Whip" and the motion of both hands to the

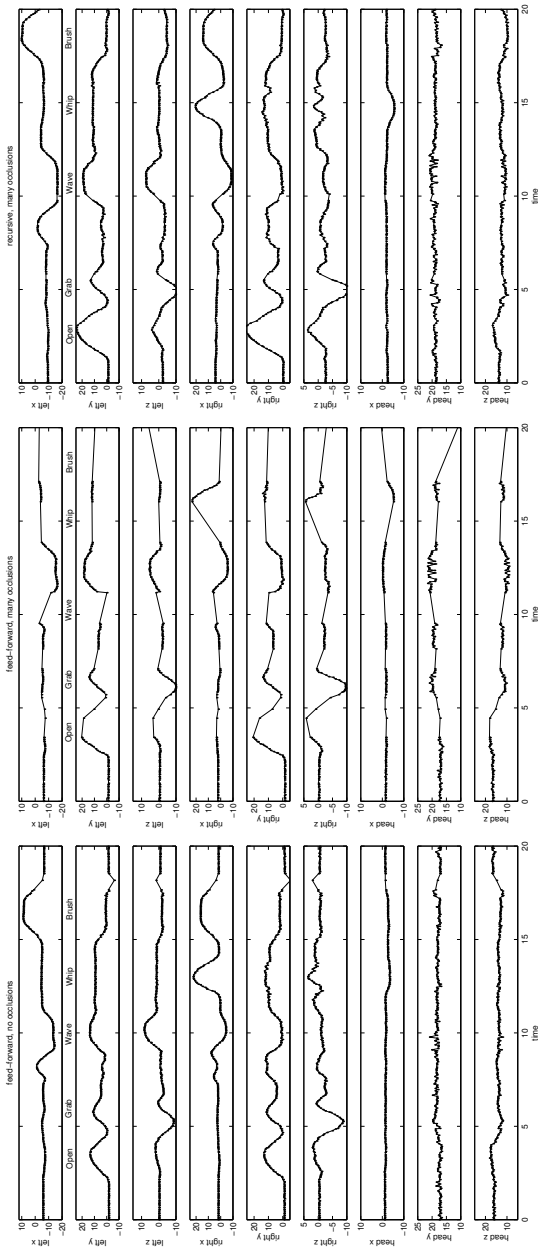

Figure 25: **Bottom**: This example is performed carefully to avoid self-occlusions. **Middle**: An intentionally pessimal example demonstrating all possible failure modes. **Top:** The recursive tracker is is able to resolve the ambiguities.

right in "Brush Knee".

The middle plot of Figure 25 shows the tracking output from the bottom-up 3-D blob tracker for a slightly different execution of the sequence. To a human observer the motions are very similar except that the motions in the failure case are slightly more exaggerated. The tracking data looks significantly different due to the tracking failures associated with occlusions. The first salient feature shows significant clipping as the top of the gesture is lost when one or both of the hands occlude the face in one or both of the cameras. This is caused by the hands being raised higher than ideal (or, alternatively if the student is shorter than the master who taught the sensei). The second feature is lost because the hands are held too close together. This causes the 2-D blob trackers to fail in one or both cameras. Almost no tracking data is acquired after 14s due to this error.

The top (right) plot of Figure 25 is the result of tracking motion very similar to that represented by the failure modes on the left. The difference is that in this case the motion is tracked by the recursive system described in Section 3. Both features are present and well-formed. This is true despite the fact that the hands were up high enough in opening form to cause occlusion with the face and close enough together to cause 2-D tracking failures at the end of the sequence. The recursive structure of the tracker allows the system to integrate information at all levels of modeling to produce good tracks in these ambiguous cases, and sensei is better able to evaluate the student's true departures from the ideal and provide more appropriate feedback.

4.2.2 Whack-a-Wuggle

Whacka is a virtual manipulation game. A virtual actor mirrors the motions of the human's upper body. The goal is for the human to touch objects in the virtual world vicariously through the virtual actor. The world contains static objects (Wuggles) and moving objects (Bubbles). Wuggles can always be whacked by returning to the same physical location since they do not move in the virtual environment, and the mapping between the physical and virtual worlds is fixed. Bubbles move through the virtual space, so they require hand-eye coordination to pop. In addition each Bubble is at a different depth away from the player in the virtual world. Due to poor depth perception in the virtual display, popping a Bubble requires the human to reach up to the approximate location of the bubble and then perform a limited search in depth.

There are a very small number of things that people do with their hands when playing Wacka. Hands are usually either resting, whacking, or popping. Whacking quickly becomes a ballistic action once the human learns where the Wuggles map into the physical world. Popping always involves performing the more complex visual-motor feedback loop required to find the Bubbles because they have random and only partially observable depth. The pace of the game is fast enough to discourage motivated players from wasting time performing extraneous gesticulations.

This context was used to test the method of alphabet selection described

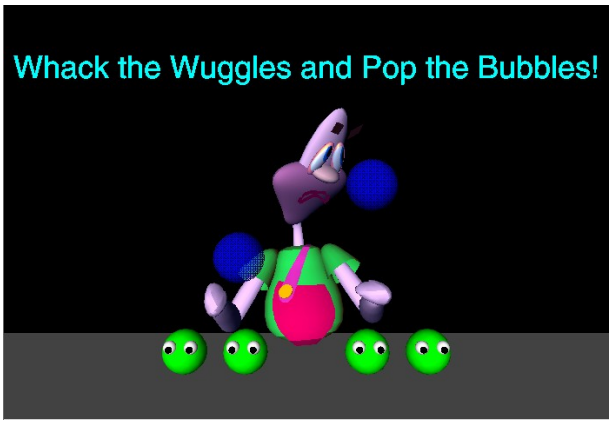

Figure 26: In Whack-a-Wuggle a clown mirrors the player's body motions. By guiding the clown's arms, the player can Whack Wuggles (the objects with eyes) and pop Bubbles (the objects floating in the air).

in Section 3.4.4. The player's motions were tracked and hand labeled for three types of behavior: *Whacking*, *Popping*, or *Tracker Failure*. Task 1 was to recognize these three classes of behavior. Task 2 was to distinguish the playing styles of different people.

The best alphabet for distinguishing the three players consisted of 3 elements, with 10 states each. Figure 27 shows alphabet traces for the three players over approximately one minute of play. These traces are the features used for player identification. Player identification performance was 75% for three players. Gesture recognition performance was also over 70%.

These results demonstrate the feasibility of recognizing gestures from model innovations. Recent work on temporal sequence clustering [45, 1, 32, 38] provides the clustering tools required for similarly attacking more complex domains.

5 Background

The implementation described above combines a rich 3-D model of the human body with probabilistic image features and a recursive formulation to realize robust tracking of the human form. DYNA also explicitly incorporates learned patterns of control into the body model. This level of modeling detail and careful feedback design greatly improves robustness and opens the door to very subtle analysis of human motion. However, not all applications require this level of detail. There is now a rich literature on human motion perception, and it is full of alternative formulations. This section will introduce you to some of the key papers and place them within the context of the DYNA framework.

Perhaps the first efforts at body tracking were by Badler and O'Rourke in 1980[34], followed by Hogg in 1988 [33]. These early efforts used edge infor-

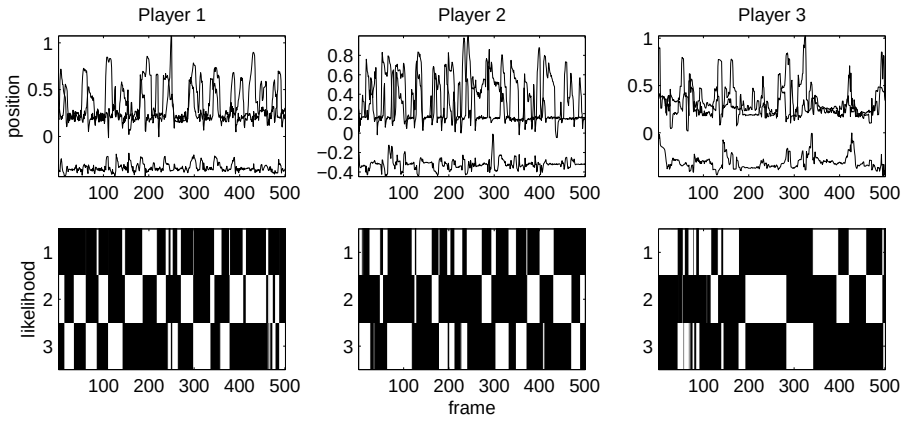

Figure 27: **Top:** Tracking data. **Bottom:** Corresponding likelihood trace of the identification alphabet.

mation to drive a kinematic model of the human body. These systems require precise hand initialization, and can not handle the full range of common body motion.

Following this early work using kinematic models, some researchers began using dynamic constraints to track the human body. Pentland and Horowitz employed non-rigid finite element models driven by optical flow in 1991[36], and Metaxas and Terzopolous' 1993 system used deformable superquadrics [27, 31] driven by 3-D point and 2-D edge measurements.

Authors have applied variations on the basic kinematic analysis-synthesis approach to the body tracking problem[42, 6, 21]. Gavrila and Davis[19] and Rehg and Kanade[41] have demonstrated that this approach has the potential to deal with limited occlusions, and thus to handle a greater range of body motions.

The DYNA framework is most closely related to the behavior modeling work of Pentland and Liu in 1995[37], Blake[24] and Bregler[8]. Those papers introduce the idea of modeling human behavior as the interaction between low-level observations and dynamic models in a recursive framework.

5.1 Particle Filtering

It is useful to compare the approach advocated here with that taken by particle filtering approaches such as the CONDENSATION algorithm proposed by Isard and Blake[25]. Both approaches have their roots in recursive filtering and control theory as described in Section 2.

The fundamental difference between recursive methods and particle filtering is in the representation of the probability density from which the state estimate is drawn. Recursive filtering methods, of which the Kalman filter is an example, represent the density parametrically. The CONDENSATION algorithm is a frame-

work for representing the density as a set of samples drawn from the underlying distribution and is therefore not parametric. The Multiple Hypothesis Testing (MHT) framework from the control and estimation community[3] provides an extension to unimodal recursive filtering frameworks that allows multi-modal, parametric representations. Work by Rehg[11] shows how unimodal trackers such as the one described here may be embedded in the MHT framework.

All recursive filtering frameworks require a mechanism for propagation forward in time of the estimated density. In Kalman filtering this is accomplished in parameter space using a unimodal model of state dynamics to update both the mean, $\hat{\mathbf{x}}_t$, and variance, $\mathbf{\Sigma}_t$, of the density. Kalman filters assume a linear model of the state dynamics, $\mathbf{\Phi}$. The mean and variance updates are accomplished with the well known closed-form equations[2].

In the CONDENSATION algorithm, the distribution is represented non-parametrically, so it must employ a different mechanism for state update. Specifically the distribution is represented as a set of samples, S_t, in model state space with an associated set of weights Π_t. The system dynamics are represented as a density conditional on the current model state: $p(X_{t+1}|X_t)$. Model update begins by selecting a sample, $s_t^{(i)}$, from the set S_t with probability determined by the set of weights Π_t. A new sample is then drawn from the corresponding density $p(X_{t+1}|X_t) = s_t^{(i)}$, and becomes the sample $s_{t+1}^{(i)}$ in the updated representation. The new weight $\pi_{t+1}^{(i)}$ is determined by comparing the sample $s_{t+1}^{(i)}$ to the image evidence.

A tracker using CONDENSATION must maintain a large number of samples to adequately represent the underlying distribution. The literature calls for approximately 1500 samples to track a bouncing ball with a single degree of freedom, and as many as 15,0000 samples to track the hand during a drawing task [26]. The advantage is that the distribution is not required to match a pre-defined parametric form. Since the Kalman filter chooses a parametric representation that is unimodal, it is possible for ambiguous observations to move that single mode away from the truth. Dynamic state updates will likely cause the location of the mode to diverge further from the truth over time. The sampled representation of the density in CONDENSATION means that it is possible for a portion of the probability mass to be allocated to divergent hypotheses. As long as there are a sufficient number of samples available to adequately represent the distribution, alternate hypotheses may be propagated forward as long as there are observations to support them.

One issue with Kalman-derived filtering techniques is that it is difficult to formulate closed-form update equations for arbitrary parametric density representations as is done for the unimodal Gaussian case. MHT is a method for maintaining a set of N unimodal Gaussian hypotheses in the Kalman filtering framework without the need to reformulate the update equations. The MHT method is essentially a bank of Kalman filters. Each Kalman filter makes predictions according to its own, possibly unique, state estimate and dynamic model. The resulting innovations sequence is a measure of how well each filter is explaining the observed data. Filters with large innovations are poor predictors and

are candidates for reinitialization [3]. Due to the fact that each filter accounts parametrically for some component of the underlying density, only a handful of hypotheses are required, as compared to the thousands of point hypotheses required by CONDENSATION. The literature shows that MHT can track a human figure, even with relatively weak underlying unimodal trackers, with as few as 10 hypotheses[11].

These methods therefore represent extremes of a continuum where computational cost can be traded off against modeling cost. For systems where the cost of model-building is very high compared to the cost of computation, CONDENSATION is a good, general method for tracking. For situations where the dynamics is reasonably well understood and the cost of computation is very high (as in the case of real-time human motion understanding), then parametric recursive filtering, possibly in conjunction with an MHT framework, is a better choice.

5.1.1 Recent Work on CONDENSATION

In the time since the first description of CONDENSATION in 1996 [24] there have been several documenting modifications to CONDENSATION that help avoid the curse of dimensionality involved in tracking systems with many degrees of freedom or with complex dynamic models. Mostly the improvements in computational efficiency of the algorithm are achieved by incorporating more sophisticated domain models. The pure form of CONDENSATION was the general tracker that discovered the dynamics from the observation at high computational cost. These extensions involve replacing flocks of hypotheses with domain-specific models that are better at capturing the true dynamics of the systems. These extensions essentially fill in the spectrum of trackers defined by pure CONDENSATION on one end, and the MHT framework on the other.

MacCormick and Blake [30] propose an extension that culls samples that represent inconsistent model states. This requires the CONDENSATION implementation to have more domain knowledge. For the task of tracking single degree of freedom motion of human bodies, this improvement allows tracking 2-D movement of head and shoulders of two people with 2000 samples. That is compared with tracking three people, two stationary and one moving with a single degree of freedom with 1000 samples.

Duetscher, North, Bascle, and Blake [14] adds the notion of kinematic singularities and end-stops. By adding significant domain knowledge they bring CONDENSATION much closer to MHT, requiring 50 samples to track arm motion with two degrees of freedom (elbow flexion and and shoulder rotation). They also track walking motion that is similar to, but less energetic than the dancing motions tracked by the MHT system in [11] with only 10 hypotheses.

5.2 Analysis-Synthesis

The analysis-synthesis approach is a widely used approximation to the general framework of recursive filtering. There are two major differences between the recursive framework advocated here and the analysis-synthesis (AS) framework: AS includes a significant feed-forward component, and lacks a model of system dynamics. Each of these issues are discussed in detail below.

5.2.1 Overview of Analysis-Synthesis

An AS tracker is composed of an Analysis module, a Synthesis module and a model of the system to be tracked. The Analysis module is composed of a set of image processing routines that extracts features from image observations. The Synthesis module renders the model state as a synthetic image composed of features congruent to those extracted by the image processing routines. The overall goal is to directly compare the model state to the observed image. An iterative process updates the model state until the best match between image and model is found.

Due to the complexity of real world image production, direct comparison is very difficult. The Synthesis module would have to generate photo-realistic images of each hypothesized configuration. Further, direct image comparison would likely be dominated by aspects of the scene that may be uninteresting: the direction of illumination for example. As a result, the comparison is done in an abstract feature space. Image processing routines are used to extract these features from the image. In practice these features are often edge features, but in principle could be any low-level feature. The model state is rendered into this feature space by the Synthesis model. The set of features produced by the Synthesis module is then compared to the set of features extracted by the Analysis module. Discrepancies between analysis features and synthesis features represent a residual to be minimized by the system. The minimization of this residual is solved using iterative gradient descent in model space. This requires a model of how variation in feature space maps into variation in model space. Since Synthesis likely involves complex non-linear components from revolute model variables, perspective projection, and other processes, this model is usually a local, linear approximation. This places constraints on the gradient descent step size and leads to the search having a high computational cost.

In the case of tracking articulated bodies or body-parts, the model usually consists of a kinematic linkage. This model can be linearized around the current estimated operating point to yield a set of Jacobian matrices relating variation in feature space to model state perturbance. These Jacobians are used to transform the measured residuals into model adjustments. The iteration process may be weighted by temperature coefficients, or filtered by other, more sophisticated, gradient decent approaches.

5.2.2 The Feed-Forward Nature of Analysis

The analysis module is feed-forward, meaning that the image processing routines (often deterministic filters) operate without access to any context that may be available in the rest of the system. This one failing has a serious impact on the stability of the system. The Analysis module does not have access to the high-level constraints coded in the system model or Synthesis module that might help resolve low-level ambiguities. There is no one place where images and high-level knowledge are brought together. The two domains are separated by the feature abstraction. This barrier makes AS systems more modular. The benefits of breaking this barrier are discussed in detail in Section 4.1.

It is important to note that merely providing prior information to an Analysis module is not enough. The implementation must have enough mathematical flexibility to incorporate the information in a meaningful way. As an example, convolution trackers often use simple Kalman predictors to improve tracking performance, but is this a truly recursive system?

Consider the normal operation of a convolution tracker: an image patch is copied from an image, possibly matted, and then convolved with an image (or a region of an image). The maximum resulting value is accepted as the new location of the feature represented by the image patch. The easiest, and most common, way to incorporate this prior knowledge into the tracker is to constrain the search to the likely area. This does not result in a better answer. It merely results in a more efficient answer since less area needs to be searched[23]. If the "right" answer is within the oval it will be found faster, but there will be no difference between the result of this search and the result of the slower search. A truly recursive system would yield a different maximum.

This raises the question of feature selection. Probabilistic features such as the blobs presented in Section 3 provide a mathematically natural method of integrating high-level knowledge. Features such as convolution patches, edges, and other deterministic, filter-based features, do not provide easy integration of context.

5.2.3 Lack of System Dynamics

The second major difference between the two techniques is the form of the model update. In AS, a distance metric is induced in the chosen feature space. These gradients are then transformed into model space to determine the appropriate system update. The system update is completely driven by observations. The Recursive Filter update is shaped not only by observation models, but also the system dynamics, as described in Section 3. This enables the Recursive Filter to take advantage of the constraints of system dynamics when determining the appropriate system update. Since real dynamic systems are heavily constrained by the laws of physics and the structure of their control mechanisms, the system dynamics is an important source of context for the observer.

The lack of these constraints on the update mechanism in AS means that past model configurations do not inform the current choice of state estimate.

Ambiguities or errors in feature extraction may result in the iteration process finding an explanation that, while plausible in isolation, is impossible due to the physical and practical limitations on the actual system being observed. One example of this sort of failure is the instantaneous flip between two solutions that are ambiguous in the absence of dynamic constraints.

5.2.4 How Real Systems Address These Shortcomings

When building an AS system, it is necessary to find ways to overcome these shortcomings in the framework. This section will discuss several systems, their contribution to AS work, and how they relate to true recursive filtering.

Gavrilla and Davis [19, 20] employ a kinematic model of the human body in their AS framework. The kinematic skeleton is augmented with a skin composed of tapered superquadrics. The subjects wear tight clothing, so the chains of tapered superquadrics are a good approximation to the external silhouette of the body. Edge energy is used as the common feature space: it is extracted from the image by a filter and rendered from the model. This, combined with the model parameter search for each image, makes the system computationally very expensive. This is an example of classic AS applied to the tracking of the human body. Later improvements included avoiding the inversion of Jacobian matrices to improve the performance of the search algorithm.

Kakadiaris and Metaxas [28] also use contour information combined with a detailed volumetric model of the human body. Unlike the above system, this system uses 3-D deformable models to track features in the image sequence. These low-level trackers represent a further dimensionality reduction, and thus undoubtedly improve the computational performance of the system. However, the trackers do not receive assistance from the body model and are subject to failures. The system partially deals with these failures by predicting occlusions. These predictions are used to select fortuitous views from the collection of available cameras in an attempt to avoid using data from ambiguous 2-D tracking results. Dynamics are used to make observation predictions, but these predictions are only used to initialize the feature search, as discussed above.

Delamarre and Faugeras [13] use the AS framework to track the human body. The silhouette is the common feature space. Silhouettes are tracked using active contours that are sensitive to optic flow and image intensity. The 3-D body model is a kinematic chain composed of spheres, parallelpipeds, and truncated cones. This model is projected into the image plane and the model is adjusted to match the observed silhouette. Since active contours are used, it is possible that prior information could help drive the contour tracker. This is a distinct advantage over AS implementations that use edge energy, or other deterministic features.

5.3 Recursive Systems

Some recursive systems have relied on optical flow for tracking. Similar to regularization work, they attempt to induce a low dimensional model as a minimum mean squared error explanation of noisy flow data. They modify the regularization by utilizing the benefits of Kalman filtering: prediction of observations and corresponding revision of the state estimate given new observations. This ideally eliminates the search inherent in the regularization framework.

Pentland and Horowitz [36] demonstrate the power of this framework for tracking 3-D and $2\frac{1}{2}$-D motions on deformable and articulated objects in video. They use modal analysis to project the flow field into relatively low-dimensional translation and deformation components. In this way they take advantage of the regularization approach to reduce the dimensionality of the optic flow data, but avoid the usual search in model parameter space. These low-dimensional signals are then used by an extended Kalman filter framework to estimate the new best estimate of pose.

The mapping of optic flow data to model influence requires the ability to predict model state and project that state into the image plane to determine proximity. This requires careful initialization. Although the framework should support prediction and resolution of ambiguities such as occlusion, those issues are not addressed in the the literature.

Bregler and Malik [8] present a very interesting recursive framework for tracking of the human body similar to DYNA that includes probabilistic feature extraction influenced by dynamic models, and the possibility for tightly coupled behavioral models. The system relies on optic flow data, but tracks blobs in that data. Unfortunately the early work on tracking focused entirely on 2-D features and 2-D models. As a result the motions tracked were all 2-D motions parallel to the camera. The dynamic models were all learned from observation. This is interesting, but also limits the system because it learns models in 2-D that are not good approximations of the real system dynamics due to the unobservability of truly 3-D motions in projective 2-D imagery.

In later work [9], the system was extended to 3-D but the extension came at the expense of the interesting recursive aspects of the earlier system. The 3-D system involved a return to more classical regularization work with the addition of clever representations for rotation that made the system tractable. Like most of work covered in this chapter the focus was on off-line motion capture, not real-time interface.

5.4 Bayesian Networks

Several systems have attempted to substitute explicit models at various levels of the tracking process with Bayesian networks.

Sherrah and Gong [44] use mean-shift trackers and face detectors in 2-D imagery to recover low-level features. In a process called *tracking by inference*, the tracked features are then labeled by a hand-crafted Bayesian Network. The simple trackers will fail during periods of ambiguity, but the network repairs the resulting mislabeling. This is accomplished by computing the most likely labeling given the observations and the network, which encodes a combination of distilled knowledge about dynamics and human habit.

Kwatra, Bobick and Johnson [29] present a similar system except that the low-level features are generated by a sophisticated static analysis method called GHOST[22] and the temporal integration modeling is concentrated into a Bayesian network. The probabilities in the network are maintained using a particle filtering method. This division is advantageous because the framework can accommodate any sort of static feature estimation. The statistical model of dynamics is weak compared to explicit kinematics and dynamics, but has the advantage of being light-weight.

Pavlović, Rehg, Cham and Murphy [35] present a system that tracks walking motion parallel to the image plane. This system dispenses with precise volumetric and kinematic models. Image correlation is used to track parts of the body. The trackers are initialized by a generalized form of MHT that consists of many Kalman filters embedded in a Bayes network. The Kalman filters employ learned dynamic models to make predictions, and the innovations process is used to drive model switching.

These systems have the advantage that the probabilistic models of dynamics and habit can be estimated from data and therefore can more easily accommodate idiosyncrasies of those datasets, but the models are soft compared to explicit kinematic and dynamic models, so that convenience can come at the cost of system fidelity.

5.5 Summary

Systems that visually track human motion fall into three basics categories: analysis-synthesis, recursive systems, and the more recent wave of fully statistical methods including particle filtering and Bayesian networks. Each of these methods has it's uses. Due to their modularity, analysis-synthesis systems are simple to build and can employ a wide range of features, but they rely on expensive searches for the synthesis component and allow features failures to propagate though the entire system. Particle filtering are an easy way to get many of the advantages of a truly recursive system with minimal modeling effort, but their lack of strong models requires massive Monte-Carlo simulations to adequately represent the underlying probability density. Fully recursive systems, such as the DYNA implementation presented above, require significant modeling effort, but this effort is rewarded with robustness, efficiency, and subtlety of analysis. Other recursive systems can allow the implementor to realize

some of these benefits with less modeling effort when the application is less demanding.

6 Conclusion

Sophisticated perceptual mechanisms allow the development of rich, full-body human-computer interface systems. These systems are applicable to desk- and room-scaled systems, theatrical devices, physical therapy and diagnosis, as well as robotic systems: any interface that needs to interpret the human form in it's entirety. In this chapter you have seen the details of a computer vision system called DYNA that employs a three-dimensional, physics-based model of the human body and a completely recursive architecture with no bottom-up processes. This system is complex, but illustrated how careful modeling can improve robustness and open the door to very subtle analysis of human motion. Not all interface systems require this level of subtlety, but the key elements of the DYNA architecture can be tuned to the application.

Every level of processing in the DYNA framework takes advantage of the constraints implied by the embodiment of the observed human. Higher level processes take advantage of these constraints explicitly while lower level processes gain the advantage of the distilled body knowledge in the form of predicted probability densities. These predictions enable more robust classification decisions. Systems which omit a model of embodiment entirely, or try to hallucinate physical constraints in the image plane will fail to capture important aspects of human motion and the overall system performance will suffer. Understanding embodiment is crucial to perceiving human motion.

Systems that rely on any completely bottom-up processing will incur performance penalties to recover from inevitable low-level errors, if it is possible to recover at all. Recursive frameworks are also crucial to perceiving human motion.

As perceptual technologies continue to improve, they will enable an environment rich with interactivity. Computers will cease to be boxes that necessitate unnatural, and sometimes painful or damaging interaction. Computers disappear into the environment, and the things we do naturally will become the primary interface. Our most important experiences are interactions with other people, and as machine perception advances, computation will finally begin to engage in that conversation.

References

[1] J. Alon, S. Sclaroff, G. Kollios, and V. Pavlovic. Discovering clusters in motion time-series data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 375–381, 2003.

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

[3] M. Athans and C. B. Chang. Adaptive estimation and parameter identification using multiple model estimation algorithm. Technical Report 1976-28, Massachusetts Institute of Technology Lincoln Laboratory, Lexington, Massachusetts, USA, June 1976. Group 32.

[4] 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.

[5] Ali Jerome Azarbayejani. *Nonlinear Probabilistic Estimation of 3-D Geometry from Images*. PhD thesis, Massachusetts Institute of Technology, February 1997. Media Arts and Sciences.

[6] 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.

[7] David A. Becker. Sensei: A real-time recognition, feedback, and training system for t'ai chi gestures. Master's thesis, Massachusetts Institute of Technology Media Laboratory, 1997. also MIT Media Lab Perceptual Computing TR426.

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

[9] Christoph Bregler and Jitendra Malik. Video motion capture. Technical Report UCB/CSD-97-973, University of California, Berkeley, 1997.

[10] Lee W. Campbell, David A. Becker, Ali Azarbayejani, Aaron Bobick, and Alex Pentland. Invariant features for 3-d gesture recognition. In *Second International Conference on Face and Gesture Recognition*, pages 157–62, Killington, VT, USA, 1996.

[11] Tat-Jen Cham and James M. Rehg. A multiple hypothesis approach to figure tracking. In *Workshop on Perceptual User Interfaces*, San Francisco, Calif., November 1998.

[12] Brian P. Clarkson and Alex Pentland. Unsupervised clustering of ambulatory audio and video. In *Proceedings of the International Conference of Acoustics Speech and Signal Processing*, Phoenix, Arizona, 1999.

[13] Quentin Delamarre and Olivier Faugeras. 3d articulated models and multiview tracking with silhouettes. In *Proceedings of the Seventh International Conference on Computer Vision*. IEEE, 1999.

[14] J. Deutscher, B. North, B. Bascle, and A. Bake. Tracking through singularities and discontinuities by random sampling. In *Proceedings of the Seventh International Conference on Computer Vision*. IEEE, 1999.

[15] 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.

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

[17] Martin Friedmann, Thad Starner, and Alex Pentland. Synchronization in virtual realities. *Presence*, 1(1):139–144, 1991.

[18] Martin Friedmann, Thad Starner, and Alex Pentland. Device synchronization using an optimal linear filter. In H. Jones, editor, *Virtual Reality Systems*. Academic Press, 1993.

[19] 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.

[20] D. M. Gavrila and L. S. Davis. 3-d model-based tracking of humans in action: a multi-view approach. In *CVPR96*. IEEE Computer Society, 1996.

[21] Luis Goncalves, Enrico Di Bernardo, Enrico Ursella, and Pietro Perona. Monocular tracking of the human arm in 3d. In *International Conference on Computer Vision*, Cambridge, MA, June 1995.

[22] I. Haritaoglu, D. Harwood, and L. Davis. Ghost: A human body part labeling system using silhouettes. In *Fourteenth International Conference on Pattern Recognition*, pages 77–82, 1998.

[23] Thanarat Horprasert, Ismail Haritaoglu, David Harwood, Larry S. Davis, Christopher R. Wren, and Alex P. Pentland. Real-time 3d motion capture. In *Workshop on Perceptual User Interfaces*, San Francisco, Calif., November 1998.

[24] Michael Isard and Andrew Blake. Contour tracking by stochastic propagation of conditional density. In *Proc. European Conference on Computer Vision*, pages 343–356, Cambridge, UK, 1996.

[25] Michael Isard and Andrew Blake. Condensation - conditional density propagation for visual tracking. *Int. J. Computer Vision*, 1998. in press.

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

[27] 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.

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

[29] Vivek Kwatra, Aaron F. Bobick, and Amos Y. Johnson. Temporal integration of multiple silhouette-based body-part hypotheses. In *IEEE Computer Vision and Pattern Recognition*, December 2001.

[30] John MacCormick and Andrew Blake. A probabilistic exclusion principle for tracking multiple objects. In *Proceedings of the Seventh International Conference on Computer Vision*. IEEE, 1999.

[31] 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.

[32] David C. Minnen and Christopher R. Wren. Finding temporal patterns by data decomposition. In *Proceedings of the 6th International Conference on Automatic Face and Gesture Recognition*, 2004.

[33] 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.

[34] 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.

[35] Vladimir Pavlović, James M. Rehg, Tat-Jen Cham, and Kevin P. Murphy. A dynamic bayesian network approach to figure tracking using learned dynamic models. In *Proceedings of the Seventh International Conference on Computer Vision*. IEEE, 1999.

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

[37] Alex Pentland and Andrew Liu. Modeling and predicition of human behavior. In *IEEE Intelligent Vehicles 95*, September 1995.

[38] Fatih Porikli and Tetsuji Haga. Event detection by eigenvector decomposition using object and frame features. In *PID*, 2004.

[39] 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.

[40] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of IEEE*, 77(2):257–285, 1989.

[41] 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.

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

[43] R. Shadmehr, F. A. Mussa-Ivaldi, and E. Bizzi. Postural force fields of the human arm and their role in generating multi-joint movements. *Journal of Neuroscience*, 13(1):45–62, 1993.

[44] Jamie Sherrah and Shaogang Gong. Tracking discontinuous motion using bayesian inference. In *ECCV (2)*, pages 150–166, 2000.

[45] Padhraic Smyth. Clustering sequences with hidden markov models. In Michael C. Mozer, Michael I. Jordan, and Thomas Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, page 648. The MIT Press, 1997.

[46] Thad Starner and Alex Pentland. Real-time american sign language recognition from video using hidden markov models. In *Proceedings of International Symposium on Computer Vision*, Coral Gables, FL, USA, 1995. IEEE Computer Society Press.

[47] Charles W. Therrien. *Decision, Estimation, and Classification*. John Wiley and Sons, Inc., 1989.

[48] Vladimir N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

[49] A. S. Willsky. Detection of abrupt changes in dynamic systems. In M. Basseville and A. Benveniste, editors, *Detection of Abrupt Changes in Signals and Dynamical Systems*, number 77 in Lecture Notes in Control and Information Sciences, pages 27–49. Springer-Verlag, 1986.

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

[51] 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.

[52] Christopher R. Wren. *Understanding Expressive Action*. PhD thesis, Massachusetts Institute of Technology, March 2000. Electrical Engineering and Computer Science.

[53] Christopher R. Wren and Alex P. Pentland. Dynamic models of human motion. In *Proceedings of FG'98*, Nara, Japan, April 1998. IEEE.