M.I.T Media Laboratory Perceptual Computing Section Technical Report No. 498

Massachusetts Institute of Technology

Department of Electrical Engineering and Computer Science

Proposal for Thesis Research in Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy

Title: Understanding Expressive Action

Submitted by: Christopher R. Wren
MIT Media Lab
20 Ames St. E15-384
Cambridge, MA 02139
USA

(Signature of author)

Date of submission: June 8, 1999

Expected Date of Completion: December 19, 1999

Laboratory where thesis will be done: MIT Media Lab
Vision and Modeling Group
Alex P. Pentland, supervisor

Brief Statement of the Problem:

User interfaces make measurements of the user and use those measurements to give the user 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 expressivity of any such interface is limited by the user's ability to overcome the reality of their bodies and perform in this arbitrary feature space.

The fact that people are embodied places powerful constraints on their motion. An appropriate model of this 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. By taking this one thoughtful step closer to the original intentions of the user, we open the door to better interfaces. Understanding embodiment is the key to perceiving expressive motion.

Chapter 1

Introduction

User interfaces make measurements of the user and use those measurements to give the user 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 expressivity of any such interface is limited by the user's ability to overcome the reality of their bodies and perform in this arbitrary feature space.

The fact that people are embodied places powerful constraints on their motion. An appropriate model of this 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, are predictable. The purposeful aspects are the direct result of a person attempting to express themselves through the motion of their bodies. By taking this one thoughtful step closer to the original intentions of the user, we open the door to better interfaces. Understanding embodiment is the key to perceiving expressive motion.

Expressive power of an interface is hard to measure. Boredom can limit and color data collection. Subjective measures of efficacy are unsatisfying. Trivial contexts can hide the power of advanced interfaces. We are building a system for interface study around the the game Netrek. As a game it provides a built in metric for success: to win. It specifies a closed world that is simple enough to be tractable, not so trivial that context in meaningless. The need to communicate abstract concepts like strategy and coordination also provides opportunities to push the limits of what we expect from interfaces. We intend to use this test-bed to explore the above claims.

Context and coordination are very important in Netrek. There are programs, called robots, that know the basics of playing Netrek, but they do not, have a very good strategic engine or any ability to cooperate with other members of their team. Marcus Huber explored the idea of augmenting these robots to include cooperation with each other and found a significant advantage over uncoordinated robots[10]. We take a different approach. With a sufficiently sophisticated interface a human should be able to add strategy, coordination and structure to the robots' activities. This symbiosis between user and robots is called the

Netrek Collective.

Chapter 2 explores some perceptual technology that we will use to build this interface. Specifically it explores the link between a perceptual system and the embodiment of the user being perceived. Chapter 3 provides more details about Netrek, a first implementation of the Netrek Collective. Finally, Section 3.3 details some directions for the Collective to grow.

Chapter 2

Dynamic Model

This chapter describes a real-time, fully-dynamic, 3-D person tracking system that is able to tolerate full (temporary) occlusions and whose performance is substantially unaffected by the presence of multiple people. The system is driven by 2-D *blob features* observed in two or more cameras [1, 26]. These features are then probabilistically integrated into a fully-dynamic 3-D skeletal model, which in turn drives the 2-D feature tracking process by setting appropriate prior probabilities.

The feedback between 3-D model and 2-D image features is an extended Kalman filter. One unusual aspect of our approach is that the filter directly couples raw pixel measurements with an articulated dynamic model of the human skeleton. Previous attempts at person tracking have utilized a generic set of image features (e.g., edges, optical flow) that were computed as a preprocessing step, without consideration of the task to be accomplished. In this aspect our system is similar to that of Dickmanns in automobile control [6], and our previous research shows that we obtain similar advantages in efficiency and stability though this direct coupling.

We will show how this framework can go beyond passive physics of the body 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 by passive physics alone. In the untrained tracker these manifest as significant structure 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.

This chapter will briefly discuss the formulation of our 3-D skeletal model in Section 2.2.1, followed by an explanation of how to drive that model from 2-D probabilistic measurements, and how 2-D observations and feedback relate to that model in Section 2.2.2. Section 2.2.3 explains the behavior system and its intimate relationship with the physical model. Finally, we will report on experiments showing an increase in 3-D tracking accuracy, insensitivity to temporary occlusion, and the ability to handle multiple people in Section 2.3.

2.1 Related Work

In recent years there has been much interest in tracking the human body using 3-D models with kinematic and dynamic constraints. Perhaps the first efforts at body tracking were by Badler and O'Rourke 1980, followed by Hogg 1988 [15, 14]. These early efforts used edge information to drive a kinematic model of the human body. These systems require fairly 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 1991 employed non-rigid finite element models driven by optical flow [16], and Metaxas and Terzopolous's 1993 system employing deformable superquadrics [12, 13] driven by 3-D point and 2-D edge measurements. Again, these systems required precise initialization and could handle a limited range of body motion.

More recently, several authors have applied variations on the basic kinematic analysis-synthesis approach method to the body tracking problem [19, 2, 9]. Gavrila and Davis [8] and Rehg and Kanade [18], have demonstrated that this approach has the potential to deal with limited occlusions, and thus to handle a greater range of body motions.

The work described in this chapter attempts to combine the the dynamic modeling work with the advantages of a recursive approach, by use of an extended Kalman filter formulation that couples a fully dynamic skeletal model with observations of raw pixel values, as modeled by probabilistic 'blob' models.

This system also attempts to explicitly incorporate learned patterns of control into the body model. The approach we take is based on the behavior modeling framework introduced in Pentland and Liu 1995 [17]; it is also related to the behavior modeling work of Blake 1996 [11] and Bregler 1997 [5]. However, this controller operates on a 3-D non-linear model of human motion that is closer to true body dynamics than the 2-D linear models previously employed.

2.2 Mathematical Framework

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 2.1. The elements of the framework are the observation model relating noisy pixel-level features to the higher-level skeletal model and vice versa, the dynamic skeletal model, and a model of typical behaviors. We will first describe the dynamic and observation models, and then the behavior model.

2.2.1 Dynamics

There are a wide variety of ways to model physical systems. The model needs to include parameters that describe the *links* that compose the system, as well as information about the hard *constraints* that connect these links to one another. A model that only includes this information is called a *kinematic* model, and can

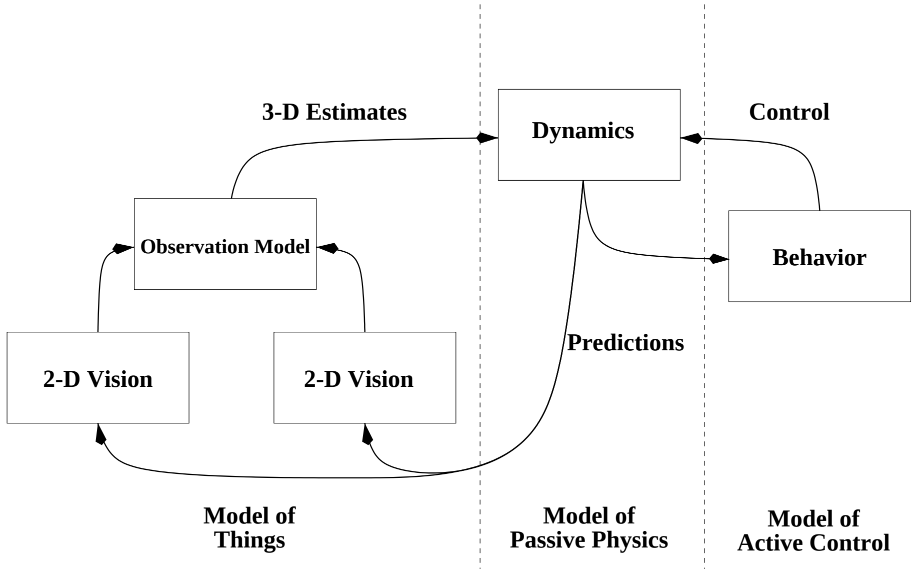

Figure 2.1: The flow of information though the system. Predictive feedback from the 3-D dynamic model becomes prior knowledge for the 2-D observations process. Predicted control allows for more sensible predictive feedback.

only describe the static states of a system. The state vector of a kinematic model consists of the model state, $\mathbf{q}$, and the model parameters, $\mathbf{p}$.

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

$$\ddot{\mathbf{q}} = \mathbf{W} \cdot \mathbf{Q} \tag{2.1}$$

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

Hard Constraints

Hard constraints represent absolute limitations imposed on the system. One example of a kinematic constraint is a skeletal joint. Our model follows the *virtual work* formulation [23]. In a virtual work formulation, all the links in a model have full range of unconstrained motion. Hard kinematic constraints on the system are enforced by a special set of forces $\mathbf{c}$:

$$\ddot{\mathbf{q}} = \mathbf{W} \cdot (\mathbf{Q} + \mathbf{c}(\mathbf{q}, t)) \tag{2.2}$$

The formulas governing these constraints can be modified at run-time.

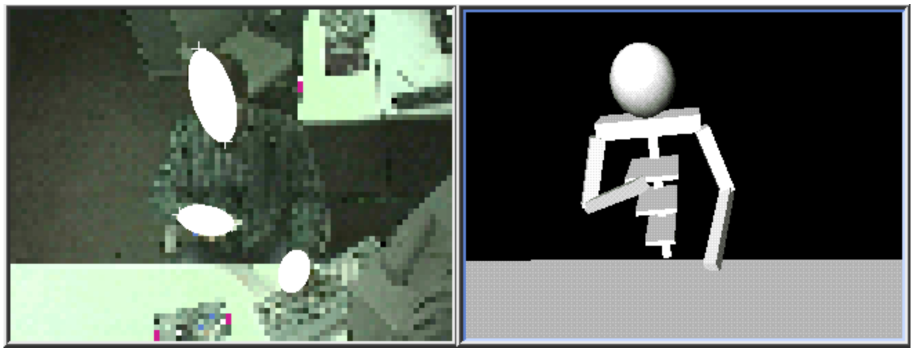

Figure 2.2: **Left:** video and 2-D blobs from one camera in the stereo pair. **Right:** corresponding configurations of the dynamic model.

It is essential that the constraint forces do not add energy to the system. It can be shown that this requirement is satisfied if they are constructed so they lie in the null space complement of the constraint Jacobian:

$$\mathbf{c}(\mathbf{q}, t) = \boldsymbol{\lambda}\frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{q}} \tag{2.3}$$

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

$$-\left[\frac{\partial \mathbf{c}}{\partial \mathbf{q}}^T \mathbf{W} \frac{\partial \mathbf{c}}{\partial \mathbf{q}}\right] \boldsymbol{\lambda} = \frac{\partial \mathbf{c}}{\partial \mathbf{q}}^T \mathbf{W}\mathbf{Q} + \frac{\partial \dot{\mathbf{c}}}{\partial \mathbf{q}} \dot{\mathbf{q}} + \frac{\partial^2 \mathbf{c}}{\partial t^2} \tag{2.4}$$

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:

$$-\mathbf{J}^T \mathbf{W} \mathbf{J} \boldsymbol{\lambda} = \boldsymbol{\rho} \tag{2.5}$$

Many fast, stable methods exist for solving equations of this form.

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.

Soft constraints such as these can be expressed as a potential field acting on the dynamic system. The incorporation of a potential field function that models a probability density pushes the dynamic evolution of the model toward the most likely value, starting from the current model state.

Note that functions that take the model state as input, such as a the controller from Section 2.2.3, can be represented as a time-varying potential field. One relevant example is incorporation of a probability distribution over link position and velocity:

$$\mathbf{Q}_f = f(\mathbf{X}, \mathbf{q}, \dot{\mathbf{q}}) \tag{2.6}$$

2.2.2 The Observation Model

The low-level features extracted from video comprise the final element of our system. Our system tracks regions that are visually similar, and spatially coherent: blobs. We can represent these 2-D regions by their low-order statistics. Clusters of 2-D points have 2-D spatial means and covariance matrices, which we shall denote $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. The blob spatial statistics are described in terms of their second-order properties; for computational convenience we will interpret this as a Gaussian model:

$$\Pr(\mathbf{O}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{\exp(-\frac{1}{2}(\mathbf{O} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(\mathbf{O} - \boldsymbol{\mu}_k))}{(2\pi)^{\frac{m}{2}} |\boldsymbol{\Sigma}_k|^{\frac{1}{2}}} \tag{2.7}$$

The Gaussian interpretation is not terribly significant, because we also keep a pixel-by-pixel support map showing the actual occupancy [24].

These 2-D features are the input to the 3-D blob estimation equation used by Azarbayejani and Pentland [1]. This observation equation relates the 2-D distribution of pixel values to a tracked object's 3-D position and orientation.

These observations 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 modeling them as descriptions of potential fields, as discussed in Section 2.2.1.

The Inverse Observation Model

In the open-loop system, the vision system uses a Maximum Likelihood (ML) framework to label individual pixels in the scene:

$$\Gamma_{ij} = \arg\max_k \left[\Pr(\mathbf{O}_{ij}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right] \tag{2.8}$$

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

To close the loop, we need to incorporate information from the 3-D model. Given the current state of the model $\mathbf{q}$, it is possible to compute the state of an individual link that matches a specific tracked feature (say the hand), and call it $\mathbf{v}$. Then, given a model of the camera, it is possible to calculate the perspective projection of that state into 2-D and call it $\mathbf{v}^*$.

Since the vision system uses a stochastic framework, it is necessary to represent this link projection as a statistical model: $\Pr(\mathbf{O}_{ij}|\mathbf{v}_k^*)$. Integrating this information into the 2-D statistical decision framework results in a Maximum A Posteriori decision rule:

$$\Gamma_{ij} = \arg\max_k \left[\alpha \Pr(\mathbf{O}_{ij}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \cdot (\alpha - 1) \Pr(\mathbf{O}_{ij}|\mathbf{v}_k^*)\right] \tag{2.9}$$

2.2.3 Models of Purposeful Motion

Observations of the human body reveal an interplay between the passive evolution of a physical system (the human body) and the influences of a an active, complex controller (the nervous system). Section 2.2.1

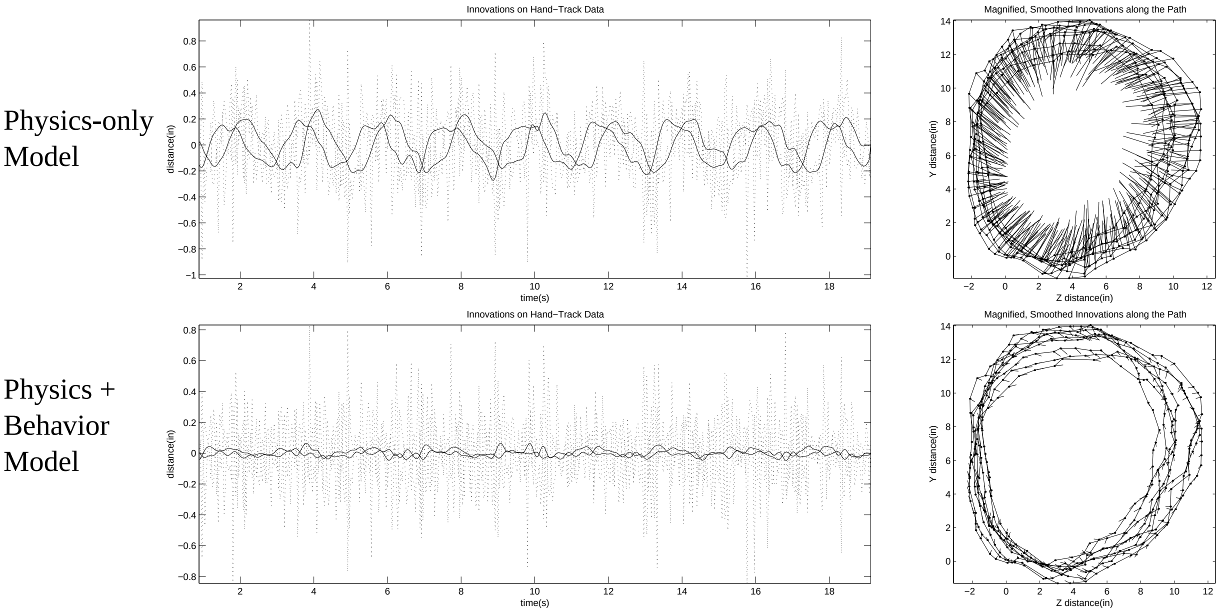

Figure 2.3: Modeling tracking data of circular hand motion. Passive physics alone leaves significant structure in the innovations process. **Top Left:** Smoothing the innovations reveals unexplained structure. **Top Right:** Plotting the Innovations along the path makes the purposeful aspect of the action 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.

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.

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 transformed by the Kalman gain:

$$\boldsymbol{\nu}_t = \mathbf{K}_t(\mathbf{y}_t - \mathbf{H}_t\boldsymbol{\Phi}_t\hat{\mathbf{x}}_{t-1}) \tag{2.10}$$

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. Assuming that the hand moves according to a linear, constant velocity dynamic model, 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 2.3 show that model is not sufficient. Smoothing $\boldsymbol{\nu}$ reveals this significant structure (top left). Plotting the innovations along the path of observations make 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 it's 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:

$$\hat{\mathbf{x}}_t = \mathbf{\Phi}_t \hat{\mathbf{x}}_{t-1} + \mathbf{I}\mathbf{u}_{t-1} \tag{2.11}$$

where $\mathbf{u}_{t-1}$ is the control signal applied to the system. The lower plots in Figure 2.3 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 2.2.1, 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.

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 [22]. 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 1993, and Isard 1996[7, 11].

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.

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[21].

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 is:

$$\Pr(\mathbf{O}_1, \mathbf{O}_2, \mathbf{q}_2 = S_j) = \left[\sum_{i=1}^{N} \pi_i b_i(\mathbf{O}_1)\mathbf{a}_{ij}\right] b_j(\mathbf{O}_2) \tag{2.12}$$

Where π_i is the prior probability of being in a 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. We restrict the observation densities to be either a Gaussian or a mixture of Gaussians. There are well understood techniques for estimating the parameters of the HMM from data.

2.3 Results

The dynamic skeleton model currently includes the upper body and arms. Figure 2.2 shows the real-time response to various target postures. The model interpolates those portions of the body state that are not measured directly, such as the upper body and elbow orientation, by use of the model's intrinsic dynamics and the behavior (control) model. The model also rejects noise that is inconsistent with the dynamic model. Table 2.4 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.

Figure 2.5 illustrates another advantage of feedback from higher-level models to the low-level vision system. Without feedback, the 2-D tracker fails if there is even partial self-occlusion, or occlusion of an object with similar appearance (such as another person), from a single camera's perspective. With feedback, information from the dynamic model can be used to resolve ambiguity during 2-D tracking. With models of behavior, longer occlusions can be tolerated.

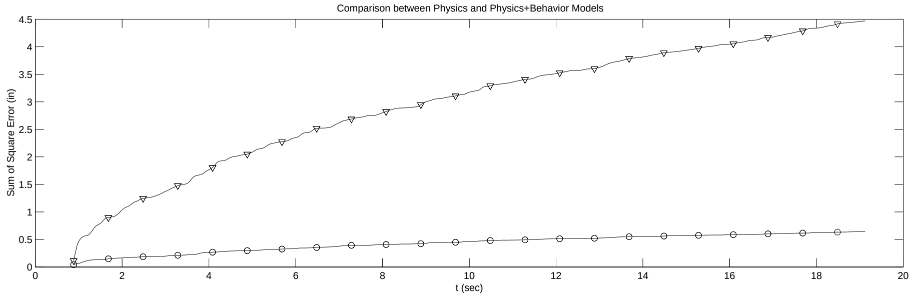

Figure 2.4: Sum Square Error of a Physics-only tracker (triangles) vs. error from a Physics+Behavior Tracker

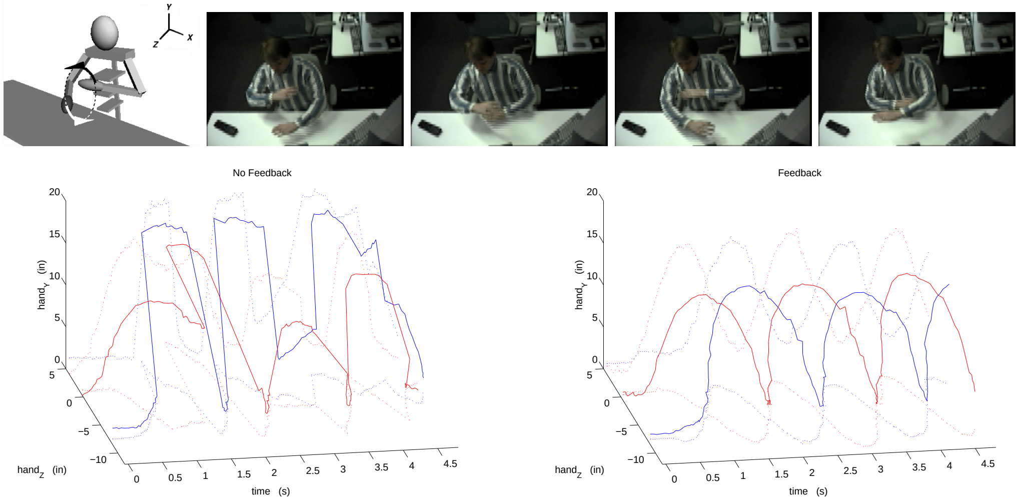

Figure 2.5: Tracking performance on a sequence with significant occlusion. **Top:** A diagram of the sequence and a single camera's view of the motion **Left:** A graph of tracking results without feedback. **Right:** Correct tracking when feedback is enabled.

2.4 Conclusion

This chapter 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 even pixel-level processes to take advantage of the contextual knowledge encoded in the higher-level models. Some of the demonstrated benefits of this approach include: increase in 3-D tracking accuracy, insensitivity to temporary occlusion, and the ability to handle multiple people.

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, our statistical models don’t have to disentangle the message from the means of expression.

The next chapter will describe a game interface built around this technology. This domain will provide opportunities to apply these techniques to a richer domain of movement.

Chapter 3

Netrek

In this chapter the game Netrek is proposed as a test-bed for perceptual user interfaces. The game Netrek provides a rich context for interfaces while retaining the closed world that makes a game environment tractable as a research platform. The next section will give an introduction to Netrek with an emphasis on the elements that make it particularly well suited to perceptual user interface work. Section 3.2 will detail the current state of the test-bed as embodied in *Ogg That There*.The last section will outline proposed additions that are intended to push perceptual user interface research and pave the way toward a truly novel human-computer interface.

3.1 The Netrek Domain

Netrek is a game of conquest with a Star Trek motif. The game is normally played by up to 16 players organized into two teams. A team wins by transporting friendly armies to each of the opposing team's planets. Figure 3.1 illustrates some of the basic elements of the game: planets, ships, and armies. The first benefit of Netrek as a test-bed for user interfaces is that it is a game: so it provides built in metrics for the success of a new interface design. If the interface allows a player to play the game more effectively (to win more), then the interface can be said to be successful with little room for argument.

Netrek is very much a team-oriented game. Winning requires a team that works together as a unit. This fact , in particular, provides a rich set of interface opportunities ranging from low-level tactics to high-level strategy. There has been some work on new tactical interfaces, but these interfaces were blocked by the Netrek community with an authentication system to keep games fair. We will concentrate on the opportunities for building interfaces for high-level communication regarding strategy since these provide the most room for novel interface design.

Netrek is usually played by groups of players on wide area networks spread over large geographic areas. The standard interface requires the user to directly control their ship. Communication with teammates is accomplished by type-written messages: this means that in order to send messages the user must temporarily give up control of their ship. So players must communicate about strategy and complex maneuvers in an

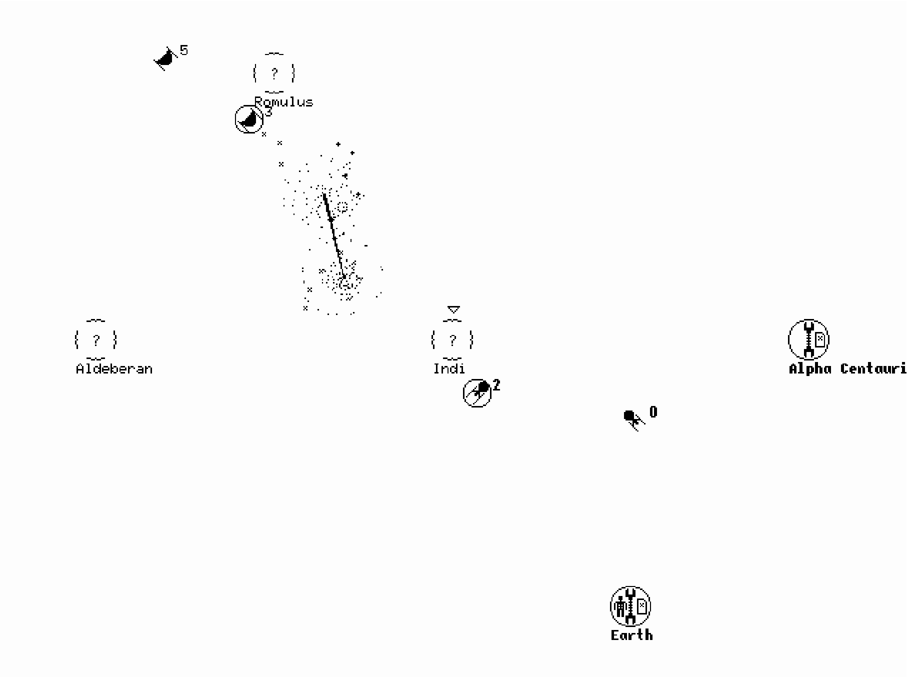

Figure 3.1: Netrek Screenshot: Federation ships $F0$ and $F2$ defend federation space near Earth and Alpha Centauri. Indi Romulus and and Aldeberan are unexplored by the Federation. Romulan ships $R3$ and $R5$ hold near Romulus. $R4$ and $F1$ have just destroyed each other near Indy.

efficient manner. This necessity led to the creation of a set of staccato jargon and an associated set of what we'll call programs. The programs are essentially little plays with a few roles that can be filled by appropriate players.

The existence of these programs is good for research in several ways. First there is pre-existing code, called a robot, that is capable of running a small set of these programs as well as game basics like tactics and navigation. These robots provide a solid base on which to build a research system. The jargon also represents a strict codification (enforced by the difficulty of communication) of human play that might indicate that recognition of plays and machine learning of plays through observation of human players would be tractable. This codification also means that there may be opportunities for novel, expressive interfaces to encourage the formation of new strategies.

One last aspect of netrek is the virtual embodiment of the robots in the game. The ships obey a simple dynamic model and they have limited control. This is particularly interesting given the proposal to represent behaviors as control signals to dynamic systems. This creates a satisfying duality between the mode of expression and the target of that expression.

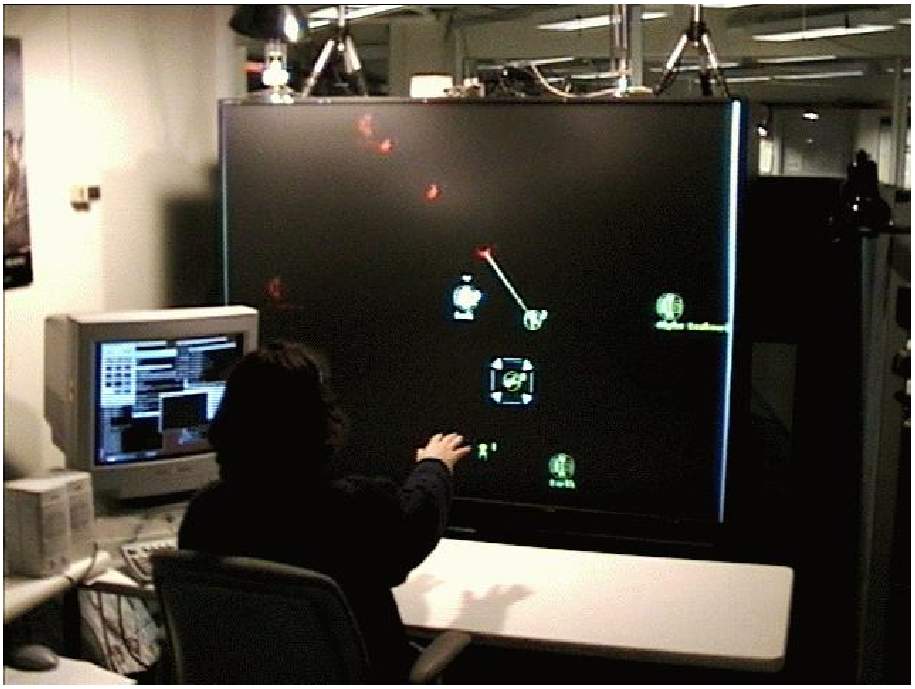

Figure 3.2: Netrek Collective Interface: the user uses a deictic gesture to select $F0$. Cameras on top of the display track the user's movements and a head-mounted microphone is used for speech recognition.

3.2 Initial Integration: *Ogg That There*

The first version of the Netrek Collective, entitled *Ogg That There*, is intended to perform in a manner similar to the classic interface demo "Put That There"[4]. Imperative commands with a subject-verb-object grammar can be issued to individual units. These commands override the robots internal action-selection algorithm, causing the specified action to execute immediately. Objects can either be named explicitly, or referred to with deictic gestures combined with spoken demonstrative pronouns. Figure 3.2 depicts a user selecting a game object with a deictic gesture.

Figure 3.3 illustrates the system architecture of *Ogg That There*. Thin, solid lines indicate standard socket-based Netrek communications. Thick, dashed lines indicate RPC based communication between our modules. The modules can be distributed across a cluster of machines to allow for future expansion of resource requirements. The following sections give details on the *Perception*, *Interpretation*, *Display* and *Robot* modules.

3.2.1 Perception: Deictic Gestures

Deictics are the only form of gesture supported by *Ogg That There*. They are labeled by speech events, not actually recognized. Interpretation of deictics relies on interpolation over a set of calibration examples obtained off-line by asking the user to point at the four corners of the screen with both hands in turn. This

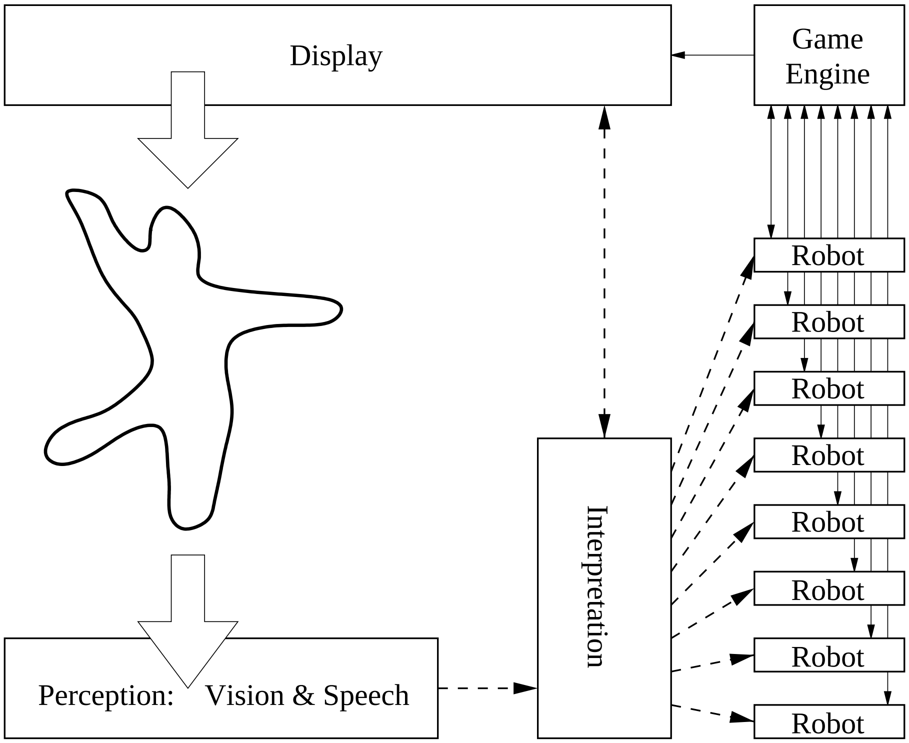

Figure 3.3: Netrek Collective System Diagram. Arrows indicate information flow. The *Display* module doubles as a database front-end for the *Interpretation* module so that no modification are needed to the *Game Engine*.

results in four sets of measurements for each hand. Separate calibrations are maintained for each hand.

In general these four points will not form parallelograms in feature space, so linear strategies introduce unacceptable warping of the output space. *Ogg That There* employs a perspective warping to translate input features ((x, y) hand position in 3D space) to screen position:

$$\begin{bmatrix} XW \\ YW \\ W \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.1}$$

where X and Y are screen position and x and y are hand position. The parameters a, b, c, d, e, f, g, h are estimated from data. With some manipulation[25] the above equation can be rewritten as a linear system of equations:

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -X_1x_1 & -X_1y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -Y_1x_1 & -Y_1y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -X_2x_2 & -X_2y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -Y_2x_2 & -Y_2y_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -X_nx_n & -X_ny_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -Y_nx_n & -Y_ny_n \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{bmatrix} = \begin{bmatrix} X_1 \\ Y_1 \\ X_2 \\ Y_2 \\ \vdots \\ X_n \\ Y_n \end{bmatrix} \tag{3.2}$$

Interpolation of a novel deictic simply involves plugging the new (x, y) into Equation 3.1. The resulting screen coordinates, (X, Y), are then passed to the *Display* which does a pick action to convert those coordinates into a game object.

3.2.2 Interpretation

The *Interpretation* module reads features from the vision system described above as well as an adaptive speech system developed by Deb Roy [20]. These feature streams must then be converted to game actions: commands to the robots or configuration of the display. The *Interpretation* module can query game state via the database engine built into the *Display* module, and can query individual robots regarding their internal state.

Currently this module is implemented as a finite state machine implementing the subject-verb-object grammar used in *Ogg That There*. The adaptive speech system is pre-loaded with a set of verbs, literal nouns and demonstrative pronouns. During the game speech events advance the state of the interpreter. If a demonstrative pronoun is encountered the interpreter resolves the gesture features in screen coordinates as described above. Those screen coordinates are then combined with grammatical constraints on valid referents from the current state of the FSM to generate a query on the *Display* database. Once a full sentence is parsed and all referents are instantiated, a command is issued to the appropriate robot.

3.2.3 Display

The state of the game is displayed on a rear-projection screen. The *Display* module generates the graphics for this screen. It is a standard Netrek client with several enhancements. An RPC interface allows remote access to standard display parameters such as viewport zoom and pan, plus addition features for the *Ogg That There* such as remote cursor display, highlighting of game objects, and textual feedback to the user. Some of these features can be seen in Figure 3.2.

The *Display* also provides a high-level interface for game information. For *Ogg That There* this is used by the interpretor to retrieve game objects that appear near a given screen location and satisfy a list of grammatical constraints.

3.2.4 Robot

The *Robot* is instantiated several times in *Ogg That There*. There is one *Robot* for each player on the game. Not shown in Figure 3.3 are the eight players on the enemy team: the same code runs players on both teams.

The *Robot* contains a large about of Netrek domain knowledge in the form of heuristic functions and implementation of several of the programs discussed above. There is also a collection of motor skills that generate commends to the *Game Engine* and allow the robot to perform all the primitive Netrek functions.

The heuristic functions are used by a primitive action-selection algorithm that selects targets and dispatches one of the tactical programs each tick. An RPC interface allows the *Interpretor* to override this system and fires a tactical program with supplied targets.

3.3 Future Work

Ogg That There is very fragile. The FSM inside the *Interpretor* limits the robustness of the system as does the rigid grammar. Limiting the user to deictic gestures denies the potential expressiveness of a gestural interface. The reliance on imperative commands issued to specific robots doesn't keep up with the pace of the game, and leads to frustration as the situation changes faster than commands can be issued. *Ogg That There* succeeded in solving many integration issues involved in coupling research systems to existing game code, but it's now time to redesign the interface to more accurately match the flexibility of the perceptual technologies, the pace of play, and the need for a game interface to be fluid and fun.

Work is already underway to replace the FSM framework with a production system that can propagate time-varying constraints against recent perceptual events to generate parses. This should alleviate much of the brittleness of the *Ogg That There* implementation. Unfortunately the state-of-the-art for production systems fall short of what we would like to have for this project. However, even if we aren't able to break out of the need for grammars, it should be straightforward to support a wider array of possible grammars as well as to recover from simple speech recognition failures. Utterances and gestures that fall outside the scope of the implemented grammars will have to be handled outside the system. Some ideas for this special handling are explored below.

Ogg That There doesn't make use of the machinery in Chapter 2 except as a means to increase tracking performance. An interpretor that made more elaborate use of gesture could provide a much richer interface. Building a gestural language system is a popular interface technique, but it requires users to be trained and distances the user from the control task by interposing a symbolic layer.

The innovations-based representations for behavior described in Section 2.2.3 combined with the embodied, physics-based, nature of the Netrek robots presents a possibility for non-symbolic communication with the robots. Innovation steams, or parametric models of the innovations, could be provided to the robots as a control strategy to be layered on top of the current task. These controls can be thought of as non-verbal adverbs that would be difficult or impossible to convey verbally. Figure 3.4 illustrates a possible situation where the user may want a carrier, $F0$, to execute a feint toward Capella before hitting the real target, Indi. Human teammates might type to $F0$, "Take Ind, feint at Cap". If the robot isn't explicitly coded to execute

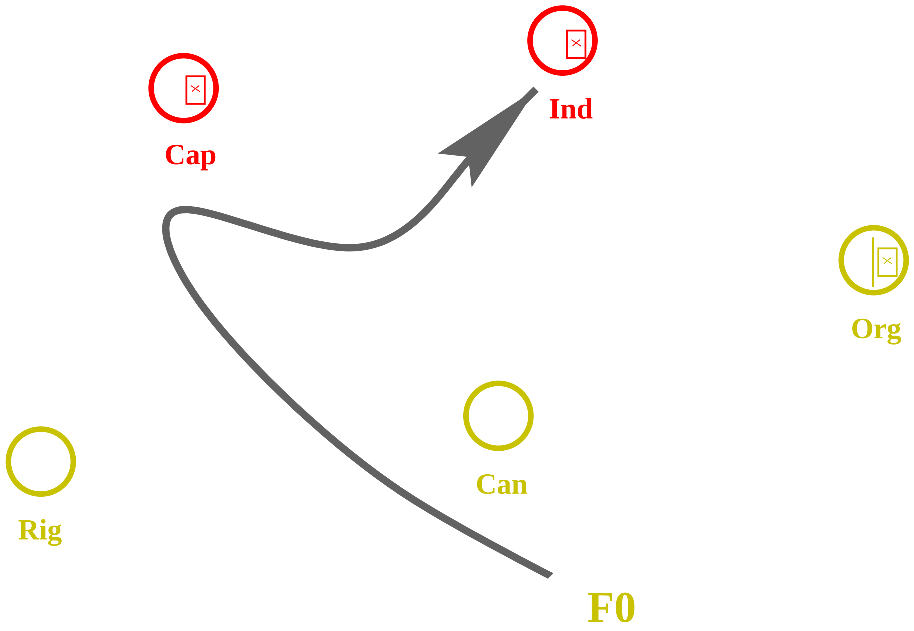

Figure 3.4: A situation where this feint path might be provided by the user gesturally even though the robot is not explicitly programmed to execute feints.

feints (or if the human player doesn't know the word feint), then this symbolic strategy will fail.

Similar strategies, with appropriate intermediate representations, may also be possible for the audio modality. A command "F1, get in there!" said in a staccato, high-energy way might bias $F1$ toward higher speeds and maximal accelerations even if the system was only able to recognize the word "F1" (or maybe not even this if the viewport is on $F1$ or the user is gesturing toward $F1$). It seems that this may end up being somewhat more symbolic since the feature space of speech is so different from the control space of the robots. An analysis system might recognize agitated speech and generate a symbol representing agitation that the *Interpretor* could choose to pass on to one or more robots.

While the preceding discussion is hypothetical, one modification to the *Robot* motor skill code has already affected qualitative changes in the robots behavior without the need to modify the existing programs that use that skill. This functionality is not used in *Ogg That There*. It allows the *Interpreter* to specify a warping of space that affects how the low-level navigation skill expresses itself. Figure 3.5 illustrates an example of how a ship might avoid an area indicated as dangerous in its way to a target. This communication channel is probably more useful for global influences, as opposed to the more local, control-level example of the feint described above.

An even more indirect, global, non-symbolic influence involves the possibility of modifying the way that the robot decides what action to take. *Ogg That There* over-rides the action-selection algorithm completely. So, either the robot is making it's own decisions or it is following the imperative commands from the user. There is no possibility to simply bias the decisions of the robot in a certain direction. Figure 3.6 shows a possible scenario where a robot chooses a different target depending on the presence of a bias indicating a difference in the value of the targets as perceived by the user.

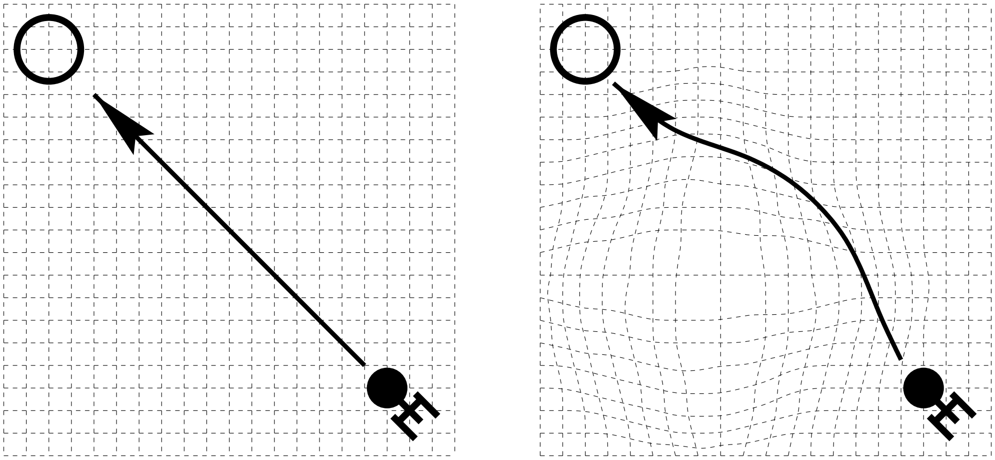

Figure 3.5: Modification of the navigation motor skill can affect a warping of space for any code that uses the low-level skill.

The current action-selection implementation is the original code that came with the robots from the Netrek community. It is a rather brittle (not to mention obfuscated) collection of heuristics. The first step toward this sort of interaction with the robots will be the adoption of a more sophisticated action-selection implementation. An obvious choice is to use the current implementation of Bruce Blumberg's reactive behavior architecture, since it has proven itself flexible and is readily available [3].

While this mode provides the least direct control over individual robots, it is important to note that this is also a mechanism for specifying high-level strategic goals and hazards that can affect many robots at once. Moving away from the imperative command structure toward a method for specifying abstract goals will increase the ability of the interface to keep pace with the game. Deictics Will Undoubtedly be important for specifying these goals, but natural specification of the polarity and severity to be associated with the demonstrated region will probably rely on stylistic attributes of the deictic and accompanying voice events. That makes this class of communication another interesting challenge.

All of these possibilities have a theme in common: they are attempting to extract content from parts of the user input that are normally ignored by classic user interface techniques like those illustrated in *Ogg That There*. An extreme example is useful to illustrate: imagine that the user foresees imminent disaster. The user does not have time to communicate in a lucid fashion, but given the desperation of the situation, they are likely to try anyway. Classical interfaces would experience speech and gesture recognition failures, and would either give up or, in the most advanced case, would ask the user a leading question. This is exactly the wrong response. There are probably only a few bits of information present in the user's desperate squeaking, but they are *very* important bits: "**look out!**" The only kind of process that is going to recover these bits is one that is attending to the nature of the signals: the energy and pitch of the voice, and the style (in an innovations-based, statistical sense) of the gesticulations.

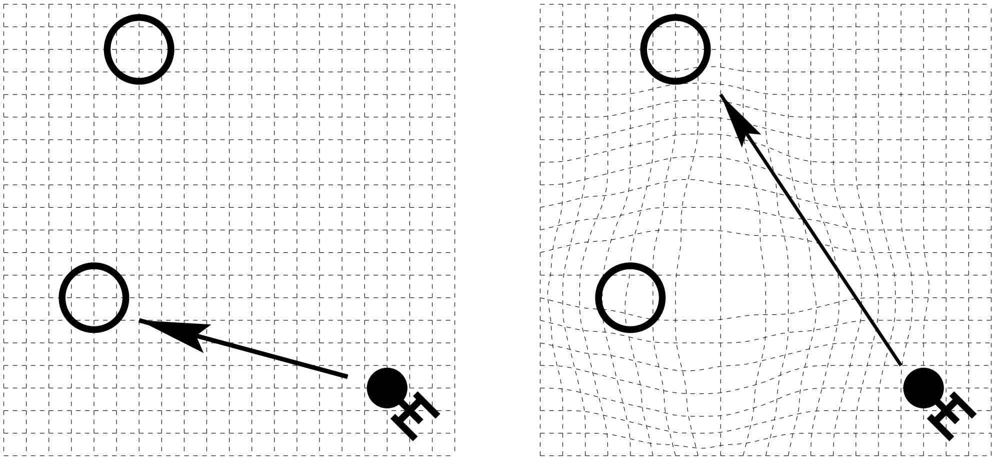

Figure 3.6: Modification of the action-selection algorithm to include user supplied weights. In this example the ship may choose to attack a different target because the most desirable target exists in an area deemed undesirable by the user.

Chapter 4

Conclusion

Understanding embodiment is the key to perceiving expressive motion. An appropriate model of this embodiment allows a perceptual system to separate the necessary aspects of motion from the purposeful aspects of motion. By getting closer to the original intentions of the user, we should have an easier time implementing the interface since our statistical tools will not have to explain parts of the signal that are easily explainable with other techniques.

Several levels of novel interface enhancements to the *Ogg That There* application are proposed. These enhancements will illustrate the power of choosing a feature set that matches the embodiment of the user. The added sophistication should increase the efficacy of the interface through increased expressiveness that would be difficult to add with classical techniques.

Bibliography

[1] 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.

[2] 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.

[3] B. Blumberg. Action-selection in hamsterdam: Lessons from ethology. In *The Proceedings of the 3rd International Conference on the Simulation of Adaptive Behavior*, Brighton, August 1994.

[4] R. A. Bolt. 'put-that-there': Voice and gesture at the graphics interface. In *Computer Graphics Proceedings, SIGGRAPH 1980,*, volume 14, pages 262–70, July 1980.

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

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

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

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

[9] 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.

[10] Marcus J. Huber and Tedd Hadley. Multiple roles, multiple teams, dynamic environment: Autonomous netrek agents. In *Autonomous Agents '97*. ACM SIGART, 1997. http://sigart.acm.org:80/proceedings/agents97/.

[11] 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.

[12] 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.

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

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

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

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

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

[18] 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.

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

[20] Deb Roy and Alex Pentland. Multimodal adaptive interfaces. In *AAAI Spring Symposium on Intelligent Environments*, 1998. also Vision and Modeling Technical Report #438, MIT Media Lab.

[21] 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.

[22] 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.

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

[24] 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.

[25] Christopher R. Wren. *Perspective Transform Estimation*. MIT Media Lab, Cambridge, MA, USA, December 1998. http://www.media.mit.edu/ cwren/interpolator/.

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