

# Model-Based Motion Filtering for Improving Arm Gesture Recognition Performance

Greg S. Schmidt\*

Donald H. House†

## Abstract

We describe a model-based motion filtering process that when applied to human arm motion data leads to improved arm gesture recognition. By arm gestures, we mean movements of the arm (and positional placement of the hand) that may or may not have any meaningful intent. Arm movements or gestures can be viewed as responses to muscle actuations that are guided by responses of the nervous system. Our method makes strides towards capturing this underlying knowledge of human performance by integrating a model for the arm based on dynamics and containing a control system. We hypothesize that by embedding the human performance knowledge into the processing of arm movements, it will lead to better recognition performance. We present details for the design of our filter, our analysis of the filter from both expert-user and multiple-user pilot studies. Our results show that the filter has a positive impact on the recognition performance for arm gestures.

## 1. Introduction

Gesture recognition techniques have been studied extensively in recent years because of their potential for application in user interfaces. It has long been a goal to apply the “natural” communication means that humans employ with each other to the interfaces of computers. People commonly use arm and hand gestures, ranging from simple actions of “pointing” to more complex gestures that express their feelings and allow communication between each other. Having the ability to recognize arm gestures by computer creates many possibilities to improve application interfaces, especially those requiring difficult data manipulations (e.g., 3D transformations). Pointing operations would certainly be an effective means to infer directional information such as where to move an object in the computer environment. To date no method has been found for arm gesture recognition that is both very accurate and extendable to different sets of gestures. Typical approaches (e.g., HMMs, neural networks) have focused on applying analytical methods for breaking down motion sequences and recognizing patterns.

The human model-based approach takes into consideration that while a person is making gestures, the resulting motions and poses are played out by a known, rather than an unknown, process. The gestures can be viewed as responses of a skeletal frame to muscle actuations that are made in response to control signals originating in the nervous system. The structure of the skeleton, joints, and musculature, is well known and well studied. The neural control systems that actuate the muscles are becoming better understood. With a solid model of human dynamics and control, much of the analytical heuristic guesswork might be eliminated. The arm is a good subject for testing

model-based approaches because it is an articulated structure with well understood musculature and fairly large inertias that must have a significant effect on gesture performance.

In our work, we have designed a filter for enhancing the signal leading to the gesture recognizer. Our motion adaptation filter integrates both physical and control models of human gestural actions into the process. The motion adaptation works by using two filters: one is augmented with a “learned” parametric gesture sequence and control system, while the other has no augmentation. Our method for incorporating process knowledge—the model and its dynamics—is the extended Kalman filter, though any process estimation filter can be used that can handle non-linearities. The squared difference between the outputs of both filters is summed and normalized, giving a score that can be used by the recognition system.

Our working hypothesis is that the motion adaptation filter will improve the unknown signal’s quality enough to improve or simplify the recognition process. We tested the hypothesis by integrating the filter with a simple template gesture recognition system, although our filter can be integrated with any standard type of gesture recognition system. We tested the system with an expert user performing multiple sets of gestures and devised a multiple-user pilot study to determine the impact that our filter has on arm-movement recognition performance.

## 2. Related Work

Here we first give an overview of the most common recognition methods and highlight their benefits and shortcomings. Then we briefly describe relevant work that utilizes human model-based approaches and related studies involving hand and arm gestures. More complete details can be found in surveys by Watson [1], Aggarwal and Cai [2] and Pavlovic et al. [3].

### 2.1. Overview of Recognition Methodologies

The common methodologies that have been used for motion and gesture recognition are: (1) template matching [1], (2) neural networks (also known as the feature-based approach) [1], (3) statistical [4, 5, 6] and (4) multimodal probabilistic combination [7, 8]. By far the most popular recognition methods are the neural networks (e.g., [9, 10, 11, 12, 13]) and the statistical method, hidden Markov models (HMMs) (e.g., [14, 15, 16, 17]). Each of these methods has a set of drawbacks which either affect their performance or limit their utilization by users. One of the major drawbacks is that they depend on user-specific training and parameter tuning.

The template approach compares the unclassified input sequence with a set of predefined template patterns. The algorithm requires preliminary work for generating the set of gesture patterns, and has poor recognition performance typically due to the difficulty of aligning the input with the template patterns [1].

\* Virtual Reality Laboratory, Naval Research Laboratory, Washington, D.C.

† Visualization Laboratory, Texas A&M University

The neural network approach works by pre-determining a set of common discriminating features, estimating covariances during a training process, and using a discriminator (e.g., the classic linear discriminator [18]) to classify gestures. The drawback of this method is that features are manually selected and time-consuming training is involved [1].

The HMM method is a variant of a finite state machine characterized by a set of states, a set of observation symbols for each state, and probability distributions for state transitions, observation symbols and initial states [4]. The state transitions, which are hidden to the observer, generate an observation symbol from each state. The basic premise of the HMM is to infer a state sequence that produces a sequence of observations. Learning the state sequence can help to understand the structure of the underlying model that generates the observation sequence. The major drawbacks of the HMMs are: (1) they require a set of training gestures to generate the state transition network and tune parameters; (2) they make assumptions that successive observed operations are independent, which is typically not the case with human motion and speech [19].

In a multimodal recognition process, two or more human senses are captured and/or two or more capturing technologies are combined together to recognize gestures. The multiple inputs are processed by a classifier, which rates the set of possible output patterns with a value based upon the likelihood of a match. The set of probabilities for each input are then combined in a manner to be able to select the most likely pattern. Many groups have explored combining speech and gesture together (e.g., Cohen and Oviatt [7, 8], Codella et al. [20], Vo and Waibel [21]).

## 2.2. Methods Utilizing Human Model-Based Approaches

Human model-based approaches integrate a model of human motion, typically approximated as a dynamic process and control system, into the process of filtering motion capture data of human movements. Model-based approaches using dynamics seems to have first appeared in Pentland and Horowitz [22] and others (e.g., [23, 24]). The method has been shown to improve tracking by helping reduce the search space required to determine the position and orientation of a tracked object for the next motion state. This includes estimating position and orientation when interference between the tracking equipment and tracked object occur [25].

Model-based approaches have been utilized by Zordan and Hodgins [26], Metaxas [25] and others for generating motion appropriate for animated characters. Badler [27] experimented with model-based approaches to simulate human-like virtual actors in a system he developed called *Jack*.

Wren and Pentland [28] applied dynamics to a 3D skeletal model of the body for a tracking application. They applied 2D measurements from image features and combined them with the extended Kalman filter to drive the 3D model. Their resulting tracking system was able to tolerate temporary image occlusions and the presence of multiple people in the tracked area. In more recent work [29] they explored the notion that people utilize muscles to actively shape purposeful motion. They were able to extract models of purposeful actions from a system they built. This reinforces the notion that there is important information

from the underlying structure of human motion that can be incorporated in the recognition process.

Rohr [30] studied human movements from image sequences by incorporating models from medical motion studies. He specifically analyzed people walking by applying measurements of the body joints and vertical displacement of the torso from a “walking” study to build a model of motion. The model was integrated with image input data using the Kalman filter to estimate 3D positions and postures of the subjects walking in the images. Others have performed similar work, including Hogg [31].

We explored the use of a model-based approach for arm motion recognition performance in earlier work [32]. We were not able to find enough evidence that the approach improved recognition performance at that time. We felt this was due to the lack of sophistication of the model and control system, which was based on a simple particle model representing the position of the wrist and its associated dynamics.

## 2.3. Arm and Hand Gesture Studies

The hand has been studied extensively for computer human interaction [33, 34, 35, 36, 37, 38, 39, 40]. However, fewer studies have been performed on gestures involving the arm in addition to the hand. Sturman [39] developed a system for recognizing gestures for orienting construction cranes. Morita et al. [41] shows how to interpret gestures from a musical conductor by tracking the tip of the wand. Baudel and Beaudouin-Lafon [42] designed an application that uses hand and arm gestures for controlling a computer presentation. Kahn et al. [43] studied pointing operations and Campbell et al. [14] performed a study involving T’ai Chi gestures.

## 3. Background

Here we give the background for methods that we utilized and integrated in the design of our filter.

### 3.1. Extended Kalman Filter

The extended Kalman filter (EKF) [44] estimates both the time sequence of states of an input data stream and a statistical model of that data stream. The EKF differs from the standard Kalman filter [45] in that it can be used to estimate a process that is non-linear and/or handle a measurement relationship to the process that is non-linear. The EKF can be augmented by a dynamic model of the system being tracked, and knowledge of the reliability of this model. Simply described, the filter is a set of time update equations that estimate the next state vector, current error covariance and the Kalman gain. The Kalman gain affects the weighting of measurement data versus the control model in determining the next state vector estimate. If the dynamic model is left out or is unreliable, the Kalman gain is high and the filter simply smoothes the input data.

The EKF’s prediction equations may be written

$$\begin{aligned} \mathbf{x}_{i+1}^- &= f(\mathbf{x}_i, \mathbf{u}_i, 0) \\ P_{i+1}^- &= A_i P_i A_i^T + W_i Q_i W_i^T, \end{aligned} \quad (1)$$

where  $f$  estimates the *a priori* state vector  $\mathbf{x}_{i+1}^-$ ,  $\mathbf{x}_i$  is the current state vector,  $\mathbf{u}_i$  is the process model vector at the current time

step,  $P_i$  and  $P_{i+1}^-$  are the current and *a priori* estimated error covariances,  $Q_i$  is the process model error covariance,  $A$  and  $W$  are the Jacobians of  $f$  with respect to  $\mathbf{x}$  and  $\mathbf{w}$ , respectively, and  $\mathbf{w}$  is a vector of random variables.

The filter's update equations may be written

$$\begin{aligned} K_i &= P_i^- H_i^T (H_i P_i^- H_i^T + V_i R_i V_i^T)^{-1} \\ \mathbf{x}_i &= \mathbf{x}_i^- + K_i (\mathbf{z}_i - h(\mathbf{x}_i^-, 0)) \\ P_i &= (I - K_i H_i) P_i^-, \end{aligned} \quad (2)$$

where  $K_i$  is the current Kalman gain,  $\mathbf{v}$  is a vector of random variables,  $h$  relates the state vector to the measurement vector  $\mathbf{z}_i$ ,  $R_i$  is the measurement error covariance, and  $H$  and  $V$  are the Jacobians of  $h$  with respect to  $\mathbf{x}$  and  $\mathbf{v}$ .

### 3.2. Lagrangian Formulation for Dynamics

The Lagrangian formulation for dynamics is particularly appropriate for articulated systems. The Lagrangian

$$L(\mathbf{q}, \dot{\mathbf{q}}) = E_k(\mathbf{q}, \dot{\mathbf{q}}) - E_p(\mathbf{q}) \quad (3)$$

is the difference between the kinetic energy  $E_k$  and potential energy  $E_p$  of the system as a function of state  $\mathbf{q}$ . The state is a set of generalized joint coordinates and its rate  $\dot{\mathbf{q}}$  is a set of related velocities. The Lagrangian formulation for the dynamics of a system is

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i, \quad i = 1, \dots, m, \quad (4)$$

where  $\tau$  is the set of externally applied forces and torques [46].

Solutions to Equation 4 can be found in closed form, which are more efficient and readily parameterizable than the open form derivations generated by the Featherstone algorithm [47], which is a very efficient rendition of the Newton-Euler approach to dynamics [48]. On the other hand, the open form derivations do have the advantage that they can be easily extended to handle large sets of joint-space configurations.

## 4. Motion Adaptation Filter

The design of our model-based motion adaptation filter is shown in Figure 1. It contains two extended Kalman filters: one augmented with a model of the human arm, dynamics update and control system, the other unaugmented and containing only the arm model and dynamics update components. The input or unknown motion sequence is passed through each filter, compared and a score is computed, which is used as output for the motion adaptation filter.

The unaugmented filter, in effect, smoothes the input motion sequence since it has no control system. The augmented filter attempts to influence the raw input motion sequence to follow a learned motion sequence. We illustrate this notion in Figure 2 by showing five different motion sequences (arc, line, wave, circle and angle) as influenced by an arc motion sequence. Each sequence starts on the right side and proceeds towards the left. The darkest grey line indicates the ‘‘influencing’’ arc sequence, the lightest grey is the input sequence, and the mid-grey is the output sequence. The images show the degree of influence that the arc has on the input sequences, which is controlled by the Kalman gain in the EKF.

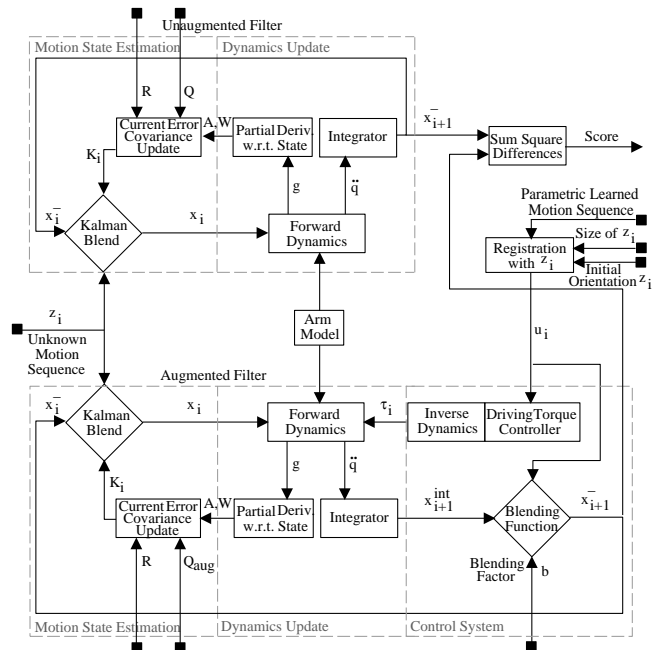


Figure 1: Motion Adaptation Filter

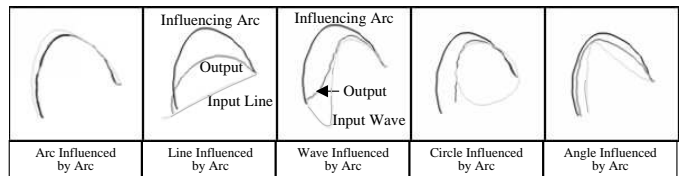


Figure 2: Five Gestures Influenced by an Arc Motion Sequence

The unaugmented and augmented filters both contain units for motion state estimation and dynamics update. The state estimation unit blends the input motion sequence with the current state vector and passes the data to the dynamics update process. There, forward dynamics are performed on the state vector producing angular accelerations. These are numerically integrated generating the next state vector. The next state vector is fed back into the system at the Kalman blend and sent to be compared with the output from the augmented filter. The Kalman gain is updated from the current error covariance which is subsequently updated by data from the dynamics update process.

The augmented filter also contains a control system, which is composed of a driving torque controller and a blending function. Torques used by the controller are derived from the parametric learned motion sequence and model and applied to the forward dynamics of the system. After numerical integration, an intermediate state vector is passed to the blending function where it is mixed with the aligned and parameterized learned motion sequence producing the next state vector. The motivation behind the augmented filter is that if the input motion sequence matches closely to the learned motion sequence (e.g., in Figure 2 the arc in arc module), then the resulting trajectory should be very similar to the input. Thus the trajectories output by the unaugmented and augmented filters will be nearly identical, and the output score will be a very small number. However, if the input motion sequence is dissimilar (e.g., in Figure 2 the line in arc module) to the learned sequence, the trajectories will differ greatly and likewise the score will be large.

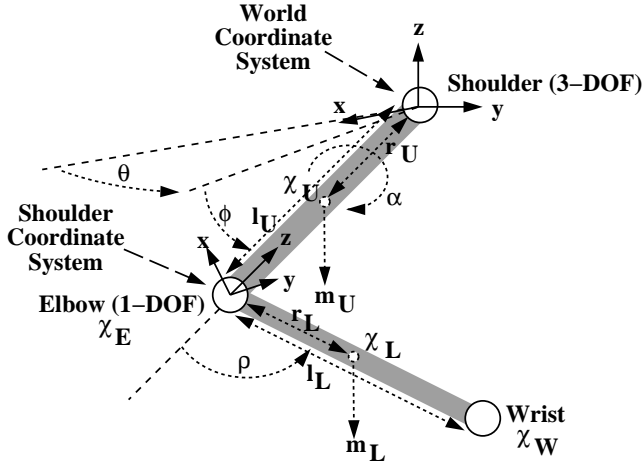


Figure 3: Articulated Arm Model

In the next few sections we describe the details of each of the components of the motion adaptation filter.

#### 4.1. Arm Model

A dynamic articulated model of a human arm is integrated into the filter. The arm model consists of a 3-DOF shoulder joint, a 1-DOF elbow joint and cylinder linkages between the shoulder and elbow, and between the elbow and wrist. The model is shown in Figure 3. We ignore the wrist twist in the lower arm. We also capture the three degrees of freedom for the torso, which is used to produce a relative coordinate system for the arm. The three degrees of freedom from the torso are eliminated after the coordinate transformation takes place between the torso and shoulder.

The position of the wrist and elbow can be determined by using the kinematics equations of motion for the arm model. The equations are parameterized using joint angles for each degree of freedom of the joints in the model. They are

$$\begin{aligned}\chi_E &= (-l_U S_\theta C_\phi, -l_U S_\theta S_\phi, -l_U C_\theta)^T, \\ \chi_W &= R_z(\phi)R_y(\theta)(-l_L S_\rho C_\alpha, -l_L S_\rho S_\alpha, -l_L C_\rho)^T,\end{aligned}\quad (5)$$

where  $\chi_E$  and  $\chi_W$  are the positions of the elbow and wrist, respectively,  $l_U$  and  $l_L$  are the corresponding lengths of the upper and lower arm,  $R_z(\phi)$  and  $R_y(\theta)$  are rotation matrices about the respective axes  $z$  and  $y$ , and  $S$  and  $C$  are sines and cosines of angles of rotation  $\theta$ ,  $\phi$ ,  $\alpha$  and  $\rho$ .

#### 4.2. Motion State Estimation

Motion state estimation is used to predict the state vector at the next time step for the current state of measured input, dynamic model and statistical models of the measured and control systems. The statistics for the measurement process and control system are in the form of error covariance matrices and are pre-determined using training and measurements from the user workspace. They are used by the EKF along with data from the dynamics update process to determine the current Kalman gain.

The Kalman gain is critical for state estimation in the system and requires knowledge from the dynamics and measurement process. This data includes the four (8x8)-Jacobian matrices

$A$ ,  $W$ ,  $H$  and  $V$  from Equations 1 and 2 which relate the process and measurement system's state vectors to the current state vector. The elements of these matrices are predetermined symbolically and updated numerically as the filter operates. They are

$$A = \begin{bmatrix} \frac{\partial g}{\partial \mathbf{q}} & 1 + \frac{\partial g}{\partial \mathbf{q}} \end{bmatrix}, \quad W = \begin{bmatrix} \frac{\partial g}{\partial \mathbf{w}_1} & 1 + \frac{\partial g}{\partial \mathbf{w}_2} \end{bmatrix},$$

and  $H = V = \mathbf{I}$  where  $\mathbf{I}$  is the 8x8-identity matrix. The matrices  $A$  and  $W$  are updated by taking the partial derivatives with respect to the current state vector of their respective complete forward dynamics equation  $g$ . The augmented and unaugmented filters have different formulations. The formulation for the augmented filter is

$$g(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{w}_1, \mathbf{w}_2) = B'^{-1}[\frac{1}{2}(\dot{\mathbf{q}} + \mathbf{w}_2)^T \frac{\partial}{\partial \mathbf{q}} [B'](\dot{\mathbf{q}} + \mathbf{w}_2) - \dot{B}'(\dot{\mathbf{q}} + \mathbf{w}_2) + \tau(\mathbf{q}^m, \dot{\mathbf{q}}^m)], \quad (6)$$

and for the unaugmented filter is

$$g(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{w}_1, \mathbf{w}_2) = B'^{-1}[\frac{1}{2}(\dot{\mathbf{q}} + \mathbf{w}_2)^T \frac{\partial}{\partial \mathbf{q}} [B'](\dot{\mathbf{q}} + \mathbf{w}_2) - \dot{B}'(\dot{\mathbf{q}} + \mathbf{w}_2)], \quad (7)$$

where  $\mathbf{w}_1$  and  $\mathbf{w}_2$  are vectors of random variables representing "white" noise with zero mean and constant variance associated with the process model's state vector and velocities, respectively.  $B$  and  $\dot{B}$  are the inertia matrices defined in Section 4.3 composed of members from the state vector  $\mathbf{q}$  and angular velocities  $\dot{\mathbf{q}}$ .  $B'$  and  $\dot{B}'$  are similar matrices to  $B$  and  $\dot{B}$  but whenever an element of  $\mathbf{q}$  and  $\dot{\mathbf{q}}$  appears, the appropriate random variable from the vectors  $\mathbf{w}_1$  or  $\mathbf{w}_2$  is added to that member. For example, if  $\theta$  appears in an element of matrix  $B$ , then it is replaced with  $\theta + \mathbf{w}_{11}$  in  $B'$ , where  $\mathbf{w}_{11}$  is the first element in the vector  $\mathbf{w}_1$ , since  $\theta$  is the first element in  $\mathbf{q}$ .

#### 4.3. Dynamics Update

The dynamics update process provides parameter updates for motion state estimation and the control system. It takes the current state of the system and the arm model (and a set of torques for the augmented filter), and performs forward dynamics to produce the parameter update functions  $g$  (described in Section 4.2) and the angular accelerations  $\ddot{q}$ . We used Euler numerical integration [49] to update the next state vector using the accelerations. We could have used a more sophisticated integration method, but we found that Euler integration was satisfactory.

Here we show the derivation of the forward dynamics equation for the 4-DOF articulated arm model, which generates the angular accelerations and is used to derive the complete forward dynamics equations (Equations 6 and 7).

In order to derive the dynamics equations, the masses, lengths and moments of inertia of the arm segments are needed. Each arm segment is represented by a thin cylinder rotating about its endpoint. The center of mass for each cylinder is estimated using data from a study on anthropometric parameters for the human body in [50]. The data gives estimations for the segmental center of mass (COM) locations expressed in percentages of the segment lengths. These are measured from the proximal end of the segments. The moment of inertia for

each segment is computed by combining the inertia tensor of the representative cylinder body and inertial component associated with the shift of its COM to the endpoint. The inertial components associated with the shift of the COM are

$$\begin{aligned}\chi_U &= (-r_U S_\theta C_\phi, -r_U S_\theta S_\phi, -r_U C_\theta)^T, \\ \chi_L &= R_z(\phi)R_y(\theta)(-r_L S_\rho C_\alpha, -r_L S_\rho S_\alpha, -r_L C_\rho)^T,\end{aligned}\quad (8)$$

where  $\chi_U$  and  $\chi_L$  are the positions in Cartesian world space of the estimated COMs of the upper and lower arm, respectively, and  $r_U$  and  $r_L$  are the corresponding radial distances from the shoulder and elbow, respectively. Time derivatives are taken to get the angular velocities at the estimated COMs of the arm segments. These are

$$\dot{\chi}_i = J_i \dot{\mathbf{q}}, \quad i = \{U, L\} \quad (9)$$

where the Jacobian matrices  $J_U = \frac{\partial \chi_U}{\partial \mathbf{q}}$  and  $J_L = \frac{\partial \chi_L}{\partial \mathbf{q}}$ , and  $\dot{\mathbf{q}} = (\dot{\theta}, \dot{\phi}, \dot{\alpha}, \dot{\rho})^T$ . The inertial components are

$$\begin{aligned}I_U &= m_U J_U^T J_U + I_{body_U}, \\ I_L &= m_L J_L^T J_L + I_{body_L},\end{aligned}\quad (10)$$

where  $I_U$  and  $I_L$  are the inertial components of the upper and lower arm, respectively,  $m_U$  and  $m_L$  are the estimated masses of the arm segments, and  $I_{body_U}$  and  $I_{body_L}$  are diagonal matrices representing the thin cylinder body inertias about each parameterized axes  $\theta$ ,  $\phi$ ,  $\alpha$  and  $\rho$ . The elements in  $I_{body_U}$  and  $I_{body_L}$  are filled by converting the cylinder's Euclidean coordinates to spherical coordinates.

The angular velocities and inertias are used to compute the kinetic energy

$$E_k = \frac{1}{2} \dot{\mathbf{q}}^T B \dot{\mathbf{q}}, \quad (11)$$

where  $B = I_U + I_L$ . The potential energy is given as

$$\begin{aligned}E_p &= -m_U g r_U C_t \\ &\quad -m_L g [l_U C_t - r_L S_r C_\alpha S_t + r_L C_t C_r],\end{aligned}\quad (12)$$

where  $g$  is the gravitational constant. The two energy terms are used for the Lagrangian,  $L$ , of Equations 3 and 4. The dynamics equations are computed and solved for angular acceleration

$$\ddot{\mathbf{q}} = B^{-1} [\frac{1}{2} \dot{\mathbf{q}}^T \frac{\partial}{\partial \mathbf{q}} [B] \dot{\mathbf{q}} - \dot{B} \dot{\mathbf{q}} + \tau], \quad (13)$$

where  $\tau$  is the set of applied torques.

#### 4.4. Control System

Our control system, in effect, acts analogously to the motor nervous system in the human body by providing guidance for how the arm model is applied to update the motion state vector. The control system specifically influences how the learned motion sequence acts on the current motion state vector. It is composed of a driving torque controller and a blending function. The driving torque controller uses data from the learned motion sequence and arm model and performs inverse dynamics, which generates torques for the dynamics update process. The blending function combines the learned motion sequence

with an intermediate state vector from the dynamics update process. The degree of its influence is controlled by a fixed predetermined blending factor. The learned motion sequence also remains fixed throughout the iteration of the filter. We see the driving torque controller as analogous to an open-loop predictive control and the blending function as analogous to proprioceptive and sensory feedback. Our control system has similarities to the model reference adaptive control (MRAC) system presented in [51, 52], which incorporates a reference model of a motion sequence, inverts its dynamics and applies the resulting torques in a controlled manner to the input data.

The torques for the driving torque controller are computed using the inverse dynamics torque formulation

$$T(\mathbf{q}, \dot{\mathbf{q}}) = \tau(\mathbf{q}^m, \dot{\mathbf{q}}^m) + \frac{1}{2} \dot{\mathbf{q}}^T \frac{\partial}{\partial \mathbf{q}} B \dot{\mathbf{q}} - \dot{B} \dot{\mathbf{q}}. \quad (14)$$

where  $\tau$  is the vector of applied torques from the controller, and joint angles  $\mathbf{q}^m$  and angular velocities  $\dot{\mathbf{q}}^m$  are from the influencing gesture sequence. The joint configurations are transformed so that they correlate with the learned model's joint configurations.

Since there is no feedback in the driving torque controller, the torques can be precomputed. When  $T(\mathbf{q}, \dot{\mathbf{q}})$  is applied to the dynamics it influences the motion of the model to follow a trajectory analogous to the influencing sequence. However, it is not necessarily strongly influencing the raw motion data to move towards the learned motion sequence. The strength of the influence is controlled by a scaling parameter  $k_c$  that is applied to the Kalman filter's process model error covariance matrix  $Q$ . This affects how much the system "trusts" the raw motion data versus the dynamic model. As  $k_c$  changes it directly impacts how the reported controller error relates to the measurement error in the system. As a result, the Kalman filter's gain matrix  $K$  (Equation 2), stabilizes differently, therefore changing how the Kalman filter weights input motion versus controller influence.

The blending function supplements the driving torque controller by providing more guidance to the state estimation. The driving torque controller provides the dynamics drive for the model, but it does not always provide sufficient guidance. The influencing motion sequence's torques may be nonlinear with respect to the joint configurations, but the tracking system performs blending of joint configurations linearly. Therefore, due to linear blending, small changes in the joint configurations can produce large changes in the dynamics. This directly affects how the driving torque controller performs. The blending function is intended to counteract this effect.

The blending function incorporates the current state of the system with the raw motion data from a learned motion sequence. The raw motion data includes the joint angles and angular velocities. This data is linear with respect to the motion state configurations of the system. The blending function that we use is

$$\mathbf{x}_{i+1} = b(\mathbf{x}_i + \Delta t \dot{\mathbf{x}}_i) + (1 - b) \mathbf{x}_i^m, \quad (15)$$

where  $\mathbf{x}_i = [q, \dot{q}]^T$ ,  $\dot{\mathbf{x}}_i = [\dot{q}, \ddot{q}]^T$ ,  $\mathbf{x}_i^m = [q^m, \dot{q}^m]^T$ ,  $\Delta t$  is the current time step, and  $b$  is the blending factor.

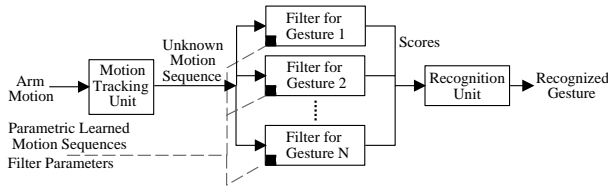


Figure 4: Test Recognition System Architecture

## 5. Analysis of Filter

In order to test the effectiveness of our new filter, we implemented it, selected a difficult-to-discriminate gesture dataset, and ran user studies.

### 5.1. Design of Test System

We designed a system to test the motion adaptation filter by adapting a simple template-style gesture recognizer. We chose the template recognition system because it is easy to implement and is very easy to understand. However, our filter can work with most standard recognition architectures (e.g., one based on neural networks). The template architecture works by comparing the unknown input sequence with each gesture pattern. For our case, the unknown input is passed through a motion adaptation filter associated with each gesture (see Figure 4 for an overview).

Human motion data is brought into the system by a motion tracking unit and segmented by searching for long pauses in the motion sequences. The choice of tracking system is arbitrary, as long it can generate a continuous sequence of motion states. For this architecture, the output is distributed in parallel to  $N$  copies of the filter. Each of the filters is custom-tuned for a specific gesture. The output of the filters is a set of scores that are processed by the recognition unit. The scores are the squared differences of the internal unaugmented and augmented filters.

In our system, we used a magnetic tracking system simply to ensure a degree of assurance that a reliable stream of input data is sent to our processing filter. There are obviously more accurate and reliable input technologies (e.g., acoustic and inertial) and vision systems, which do not guarantee the continuous reliable stream of input, but do have the potential to produce more accurate tracking. We emphasize that our filter can accept tracking data from any motion capturing technology.

Our system captures orientations of the lower arm, upper arm, and torso and returns four Euler angles. Angular velocities are estimated from the angles using time difference methods. The set of angles and angular velocities makes up a motion state vector. The sequence of state vectors is sent to the motion state estimation unit.

### 5.2. Selection of a Hard-to-Discriminate Gesture Dataset

Our first step for analyzing the performance of the filter was to select a set of gestures that are hard to distinguish from each other. The selection criterion was determined by observing the physical trajectories at the wrist position for each gesture while being performed by a user. The wrist trajectories for the gesture dataset we selected for the introductory experiments are shown in Figure 5. This gesture set was chosen because of

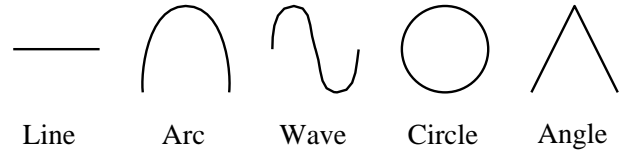


Figure 5: Wrist-Trajectory Shapes of the Gesture Datasets used for the Expert User Experiments

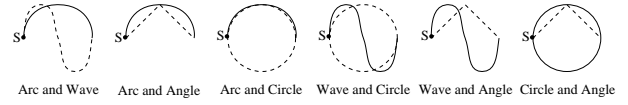


Figure 6: Overlapping Features Embedded in Gesture Pairs

the overlapping features embedded in many of the pairs of gestures (see Figure 6 for an illustration). Two distinct gestures that have overlapping motion segments, especially if they start with the same motion sub-sequence, are more difficult to distinguish than dissimilar nonoverlapping gestures. A properly tuned EKF bases its initial output more on the input data than the dynamic model. But, when it converges to a stable blending state, the dynamics of the system takeover. If two gestures have similar starting trajectories and abruptly change after the dynamics become more dominant, the system will initially fail to discriminate between the two gestures because the derived dynamics of the system are similar. Eventually the mixture of the two dissimilar segments of the gestures will influence and change the system behavior.

For our experiments, we considered the direction in which the motion trajectory was performed by the user as a means to distinguish the gesture sequences. Thus the five basic shapes shown in Figure 5 produce ten different gestures (two for each shape). We used combinations of the five basic shapes to generate gesture datasets and test the performance, generalizability and extensibility of the filter in the first four expert-user experiments.

### 5.3. Filter Parameters

Our filter requires a set of parameters that must be predetermined and tuned for individual gestures. The EKF requires error covariance data for the measurement and control processes. The dynamics update requires measurements from the user's arm. The control system requires a blending constant and the learned motion sequence.

#### 5.3.1. Parameter Determination

To compute the measurement error covariance we affixed three motion tracking receivers in the user workspace to a stationary configuration analogous to that of the right arm. We recorded 1000 samples continuously and estimated the error. The error covariance matrix is computed using the angles and angular velocities. The angular velocities are estimated by treating the set of samples as a continuous stream of data and taking time differences with respect to the sampling period. The measurement error needs to be computed once for a given set of hardware and workspace.

The control process error is computed by using the pre-recorded gesture sequences. A parametric learned motion se-

quence for each gesture type is selected by determining the closest fitting trajectory to a normal trajectory that is computed from the sample set of gestures. The error matrix is estimated using the mean squared error between the parametric learned motion sequence and the rest of the sequences. The control error needs to be computed for every gesture sequence.

### 5.3.2. Subject Measurements

Some of the parameters needed for the filters are taken from measurements of the users. The filters require the lengths, radii and masses of the upper and lower arm. These parameters are obtained by combinations of two methods: direct measurements and estimation from anthropometric parameters of the human body. The lengths are determined by either directly measuring the distance between the shoulder and elbow, and elbow and wrist, or estimating them from the height and sex of the user. Estimations of anthropometric parameters of the human body are made according to the procedure outlined in Hall [50]. The radii are obtained by measuring the circumferences of the arm segments at the midpoint. The masses are obtained by weighing the subject and estimating the arm segment masses based on a study with mass measurements taken from cadavers. The masses for the arm segments are determined as percentages of the whole body mass for males and females.

### 5.3.3. Parameter Tuning

In order to use the EKF, specific parameters have to be tuned in order to get desirable guidance in the recognition units. One of the parameters that needs tuning is a multiplicative factor  $k_{aug}$  used to scale the augmented filter’s control error covariance. There is one such scaling factor for each control error covariance matrix. The scaling factor is used to adjust the level of “trust” in the filter by changing the control error with respect to the measurement error. The larger  $k_{aug}$  is, the more the filter output depends on the input. The smaller  $k_{aug}$  is, the more the filter output depends on the controller and dynamic model. As a result the Kalman gain matrix, essential for the Kalman blend, changes.

For the unaugmented filter, we used a multiplicative scaling factor  $k$  to adjust how much smoothing is performed on the input gesture sequence. This parameter, again, is applied to the control error covariance.  $k$  acts in the same manner as  $k_{aug}$  does for the augmented filter.  $k$  is adjusted by lowering its value until the input data trajectory becomes smooth and still follows roughly the same trajectory. If it is decreased too much, the output trajectory will follow the unguided dynamic model too much.

Another parameter to be tuned is the blending factor  $b$ . This is applied in the blending function, which performs a blend of the intermediate state vector  $\mathbf{x}_{i+1}^{int}$  and the parametric learned motion sequence. This factor is important because it weights how the raw data is blended with the parametric learned motion sequence at the motion state level. The Kalman blend does not directly incorporate knowledge of the parametric learned motion sequence. We used one blending factor for all the gesture types.

To show how we made our choice of parameters for the augmented filter, we devised a simple experiment that showed the

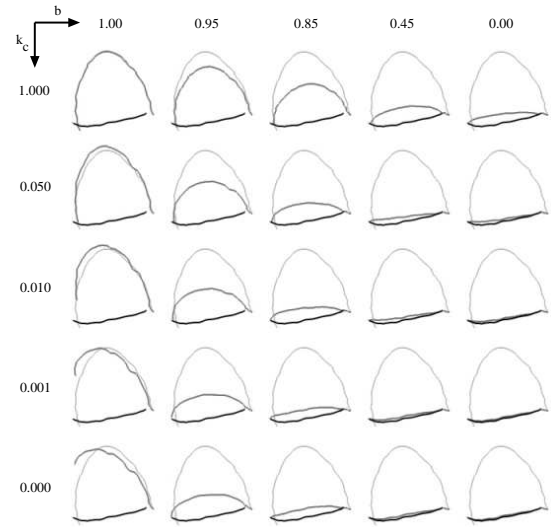


Figure 7: Effects of Varying Augmented Filter Parameters

effects of adjustments to the parameters  $k_{aug}$  and  $b$ . For demonstration purposes we chose an arc gesture as our raw data and a line as the learned motion sequence. The arc and line sequences refer to the complete motion of the arm as it traces out a spatial arc and line, respectively, at the wrist. In Figure 7 we illustrate the effect on the wrist position, over the full motion sequence, of varying the parameters  $k_{aug}$  and  $b$  from 0 to 1. In the figure, the light line is the input arc sequence, the black line is the line sequence and the dark grey line is the adapted motion.

The figure shows that the changes are not necessarily linear with respect to either parameter. In addition, both parameters produce different effects in the adapted trajectory. The controller scaling parameter appears to influence the raw motion data to morph into a rough form of the influencing sequence, but it does not align very well with it. This is evident as you look at the images in the left-most column. The blending factor appears to blend the two trajectories fairly well by itself, but not completely. An appropriate combination of the two parameters produces the best blend. For the different gestures we used varying values for  $k_{aug}$ , but we chose a fixed value for  $b$ . We tried to make our selection based on the graphs in the figure.  $b$  was chosen from the graphs that produced a mixed trajectory with a bit more influence from the learned motion sequence.

An important consideration when selecting the parameters is the degree of alignment of the input gesture with respect to the learned gesture. In the experiments, we ask the users to extend their right arm in a perpendicular direction to the front side of their body. The gestures they are asked to perform are then centered around that hand position as best as possible. Rough alignment and scaling is applied to the parametric learned gesture in addition to the parameterizing that is necessary to perform a matching comparison. This is the registration phase, which can be seen on the right side of the filter diagram in Figure 1. If the parametric learned gesture does not align very well with the gesture it is supposed to accept, it creates a high score for the comparison. This is due to our method for evaluation which compares the augmented and raw input trajectories. If the alignment is extremely bad we could not adjust the  $k_{aug}$  parameter to “trust” the model as much. In most cases this is not a problem,

but for a difficult dataset to recognize, such as the basic five gestures in Figure 5, some gestures will be improperly classified.

### 5.3.4. Sensitivity Analysis

If we were to run a full user study on human subjects of widely varying mass and height, it would be important to understand how much of an impact parameter changes have on the dynamics of the system. If it can be shown that the system is relatively insensitive to changes in the parameters then it may be considered to be more generalizable and potentially more powerful. We analyzed the sensitivity of a few of the body parameters (summarized in Schmidt [53]), but did not determine enough meaningful information to make conclusions about the generalizability of our filter.

## 5.4. Expert User Experiments

We set out to verify the effectiveness of the filter integrated into a gesture recognizer by devising a set of experiments to be performed by an expert user. These were designed to test the performance of the recognizer with and without our filter. We also wanted to ascertain something about how generalizable and extensible our filter is with respect to different and larger gesture datasets, respectively. To accomplish these goals, we ran five experiments. Before beginning we pre-recorded a database of gestures from the user, computed the parameters and learned models, and performed manual parameter tuning.

### 5.4.1. Accuracy Performance

The purpose of the first experiment is to determine the performance rating of the recognizer integrated with and without our filter. We used the five gestures from Table 1, and recorded 100 samples for each gesture. The gestures were first aligned with the learned motion sequences, then the learned motion sequences were parameterized to match the size of the input sequence. We supplied both the filtered (our method) and unfiltered recognizers with the 500 gestures. The results are given in Table 1.

Table 1: Results of Experiment #1

Arc	Line	Wave	Circle	Angle	Totals
99/100 99%	99/100 99%	100/100 100%	100/100 100%	99/100 99%	Unfiltered Approach 99.4%
98/100 98%	100/100 100%	100/100 100%	100/100 100%	99/100 99%	Our Filtered Approach 99.4%

They show that both methods have an accuracy rating of 99.4%. The fact that both methods produced acceptable results turned out to be only coincidental for the unfiltered approach, which was later shown to be very inconsistent. We analyzed this dataset further and noticed that the gestures were fairly spatially regular with respect to each other. For example, there was not an extensive amount of variation due to alignment, skewing and scaling among the like gestures in this set.

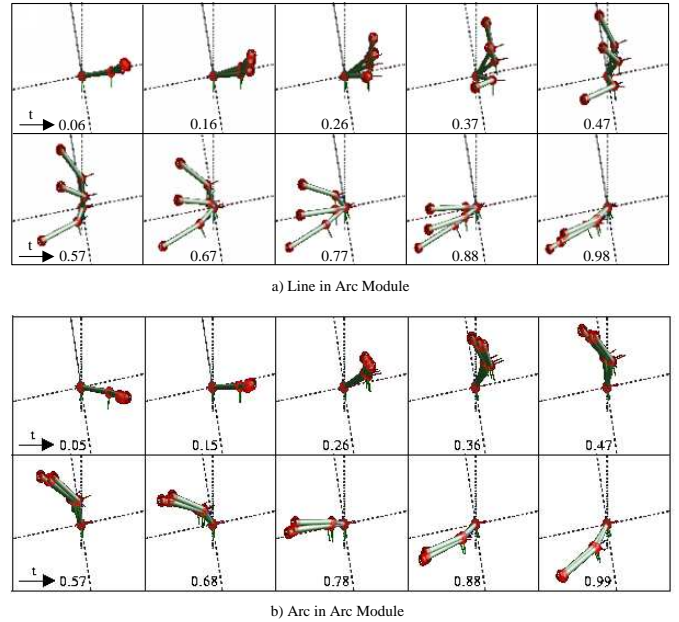


Figure 8: Arm Model Motion in Time

To get a better idea of how our method works, refer back to Figure 2. The arc in the arc module shows the best match between the augmented and the unaugmented (effectively the learned motion sequence) trajectories. The rest of the cases show that the learned arc sequence has a large influence on the data running through the augmented filter which is evident by the output augmented trajectories. This effect pulls the augmented and raw data curves apart. The sequences in Figure 8 illustrate a small set of state transitions from the three arm models used in generating the trajectories for the line and the arc in the arc module. The figures show frames from a 3D simulation of the corresponding schematic 4-DOF arm models. The arm states are very similar for the arc in the arc module, but very different for the line in the arc module.

### 5.4.2. Generalizability

To test the generalizability of our approach, we ran a second experiment. In the experiment we used the reverse-order wrist trajectories for the gestures used in the first experiment (a completely unique dataset). We recorded 100 samples for each of the five gestures and purposely added noise into the samples to test the robustness of our filter. Then we passed them into the gesture recognizer twice, with and without our filter in the system. The resulting performance ratings are given in Table 2.






In this case, the accuracy of the recognizer integrated with our filter proved to be far superior than without it. The performance rating for our filtered approach is 98.8%, while the unfiltered is 79.8%.

### 5.4.3. Extensibility

For the third experiment, we examined the extensibility of our approach. To do this, we increased the number of distinct gestures that the recognizer had to distinguish. We decided to use the two sets of gestures from the first two experiments and combine them into one database. Although diagrams make the two gesture sets appear similar, the motions that the human subject













Table 2: Results of Experiment #2

 Arc	 Line	 Wave	 Circle	 Angle	Totals
60/100 60%	100/100 100%	78/100 78%	62/100 62%	99/100 99%	Unfiltered Approach 79.8%
98/100 98%	100/100 100%	100/100 100%	100/100 100%	96/100 96%	Our Filtered Approach 98.8%

has to perform with the arm are totally different. When we performed the same experimental procedure as before we obtained the results in Table 3.

Table 3: Results of Experiment #3

 Arc	 Line	 Wave	 Circle	 Angle	 Arc	 Line	 Wave	 Circle	 Angle	Totals
99/100 99%	99/100 99%	100/100 100%	100/100 100%	99/100 99%	60/100 60%	100/100 100%	78/100 78%	62/100 62%	99/100 99%	Unfiltered Approach 89.6%
98/100 98%	100/100 100%	100/100 100%	100/100 100%	99/100 99%	98/100 98%	100/100 100%	100/100 100%	100/100 100%	96/100 96%	Our Filtered Approach 99.1%

Here our method has an accuracy rating of 99.1% while the unfiltered approach has a rating of 89.6%. This gives us a good indication that our method is extensible to larger size gesture datasets.






#### 5.4.4. More Generalizability Experiments

At this point we decided to revisit the first experiment with the hope of making it more difficult to distinguish the gestures than before. The goals of the new experiment were to show more generalizability with our method. In order to do this, we replaced the line and the wave with a triangle and another form of the arc. The new arc gesture is generated using a bend at the elbow instead of the straight arm motions used for the original arc. By our definition of arm gestures (i.e. movements of the arm that may or may not have any meaningful intent) and our analysis of only the “end-effector” position of the arm at the wrist, we do not make any distinction between the new and old arc gesture since both have identical wrist trajectories. The triangle gesture resembles the angle gesture in the first time steps, but deviates from it near the end. Our assumption was that this choice of gestures would be harder to discriminate. 75 trials were run for each gesture. The approximate gesture shapes and results of this experiment are given in Table 4.

The results show that the new gesture set was a bit harder to recognize by both methods. But our filtered approach showed an accuracy rating of 98.1% compared with the unfiltered approach’s rating of 95.2%. The results were again encouraging with regard to our method’s consistency and accuracy, and also that it generalizes to different gestures quite well.






For our fifth experiment we ran 50 trials with five new gestures, each significantly different from the others. In addition, we decided to make a choice of somewhat natural gestures. The

Table 4: Results of Experiment #4

 Arc	 Triangle	 Bent arm Arc	 Circle	 Angle	Totals
75/75 100%	68/75 90.7%	65/75 86.7%	75/75 100%	74/75 98.7%	Unfiltered Approach 95.2%
74/75 98.7%	74/75 98.7%	72/75 96.0%	74/75 98.7%	74/75 98.7%	Our Filtered Approach 98.1%

goal of the experiment was to determine if our method works well with gestures that are very easy to distinguish because they are quite distinct and are more natural. Our choice of gestures included the “zorro” sign, Catholic cross, salute, a stop and a waving gesture. Diagrams of the motions of the wrist and results of the experiment are shown in Table 5.

Table 5: Results of Experiment #5

 Zorro	 Catholic Cross	 Waving	 Stop	 Salute	Totals
50/50 100%	46/50 92%	50/50 100%	50/50 100%	50/50 100%	Unfiltered Approach 98.4%
50/50 100%	50/50 100%	50/50 100%	50/50 100%	50/50 100%	Our Filtered Approach 100%

The results show that our method was 100% accurate on this gesture set, while the unfiltered approach achieved an accuracy rating of 98.4%.

#### 5.4.5. Discussion

In the experiments, we evaluated the accuracy performance, generalizability and extensibility of our filter when integrated in a recognition system. We made steps to ensure that it was difficult to distinguish among gestures by carefully selecting gesture datasets with overlapping motion traits. When compared with the recognizer with no filter attached, our method showed improved recognition performances. Our results from the five experiments show that our method is consistently accurate with rates ranging from 98.1% to 99.4% and extends to multiple gesture datasets. This compares very favorably with the unfiltered method whose accuracy ranged from 79.8% to 99.4%.

## 6. Pilot Study

We performed a pilot study involving six different subjects, in order to evaluate our model-based approach in a more general sense (i.e. across different subjects). We also ran a followup study to test how we can reduce the amount of parameter tuning that is required for each of the filters. The details of each experiment are given in the next sections.

## 6.1. Subject Selection

For the experiment, we selected three males and three females, with varying anatomical proportions. The sex discriminant was desired to accommodate for potential unequal mass distribution that exists between male and female subjects in the arm based on muscle and bone proportions. The proportions we were concerned with were the lengths, radii and masses of the upper and lower right arm. The human subjects were selected without regard to ethnicity, age, social or cultural backgrounds. The only screening requirement we had was a visual observation of size proportions in order to assure a subject pool of varying anatomical proportions.

## 6.2. Subject Measurements and Setup

Each experiment proceeded by taking physical measurements of the subjects. Their body weight was measured and used to estimate the masses of the upper and lower right arm. The lengths of each arm segment were obtained by measuring the distance from the shoulder to the elbow and from the elbow to the wrist. The radii of the arm segments were approximated from measurements of the circumference of the middle of the upper and lower arm segments.

The subjects had body weights ranging from 55 to 87 kg and heights ranging from 1.6 to 1.9 m, giving us a broad spectrum of masses and lengths for the user’s arm proportions. The upper arm lengths varied from 28 to 36 cm and the lower arm lengths from 23 to 27 cm. The upper arm radii varied from 3.66 to 5.25 cm and the lower arm radii from 3.18 to 4.38 cm.

Next we attached two motion tracking receivers to the human subjects on their right arm, using velcro straps at the wrist and near the elbow. A third receiver was affixed to their shoulder with tape.

## 6.3. Pilot Experiment 1

In the first subject experiment our goal is to compare the difference between augmenting the recognition process with a model versus not augmenting the process. The subjects were asked to perform 25 trials of each of five different gestures, using the right arm. In between each set of trials for one gesture, the subject was given ample rest time to help avert any fatigue associated with the repetitive motions they were asked to make. We used the same five gestures as illustrated in Table 5, the zorro, Catholic cross, stop, salute and waving gestures.

The results we obtained were measurements of how well each recognition system predicted the correct gesture sequence. The performance rating for each method—the unaugmented and our model-based approach—were computed by averaging the performances for each of five different gestures. The performance for each gesture was computed by averaging the results from each of the six subjects. We then made a histogram chart (see Figure 9) comparing the two sets of data.

The data for each user was analyzed by setting the body parameters for the recognizer to their measurements before running the accuracy tests. The rest of the parameters for the recognizer were individually tuned for each subject. The results for our model-based approach show an acceptance rate of 98.7% with standard deviation of 1.0%. The unaugmented approach performed at 93.5% acceptance rate with standard deviation of

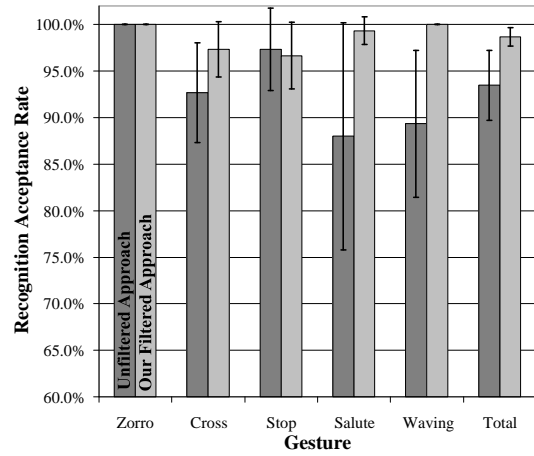


Figure 9: Comparison of the Unfiltered and Filtered Approaches

3.7%. The high acceptance rate and low variability that our results show give us a fairly good indication that integrating our filter into the recognition process improves the recognition accuracy.

A drawback of this experiment is that a significant amount of custom parameter tuning was performed for each subject. As a result, we decided to evaluate whether or not our methodology would allow us to reduce the tuning effort required by each experiment. We ran a second experiment to test this idea.

## 6.4. Pilot Experiment 2

The second experiment was performed to test whether or not custom parameter tuning for different users is necessary. The tuning phase is used to adjust the parameters for the Kalman filters and blending functions. This is a common issue with recognition methods based on neural networks, since they require training for each individual user. We decided to select the parameters from one subject, tune them, and use those settings for each of the other five subject’s tuning parameters. When we ran the experiments, the results did not prove to be as good as when individual parameter tuning for each filter was performed. However, in some cases, for a small set of the gestures, we were able to obtain normal results ranking above 98%.

Although, the results for this experiment were disappointing, our method shows great success when tuning is made for each individual user. Addressing the issue of avoiding the need for individual subject parameter tuning is an important topic for future research.

## 7. Discussion and Conclusions

We have developed a new model-based filter that incorporates a dynamics model, a control system and motion state estimation and applied it to the gesture recognition process. The dynamic model gives us a way to represent the underlying mechanical motion of the human arm. The control system acts as a means to exert control over and provide guidance for the motion applied by the dynamics. This is analogous to the human motor nervous system. The control system was aided by a blending function that supplements the driving torque controller, giving it better guidance at the motion state level instead of at the torque level.

In the future, we would like to reduce the use of the blending function and increase the sophistication of the model and controller. This would show that the dynamics are acting solely upon the recognition performance of input coming through our filter.

Our filter proved to be effective in improving the performance of the recognition process as shown by our expert-user and pilot user studies. We showed this by comparing an unfiltered recognition process with one augmented with our model-based filter. Our method works acceptably well for hard-to-distinguish gesture sets and even better for very dissimilar sets. The results definitely warrant further user evaluation studies.

Our method does involve a small amount of parameter tuning and training for the error covariances. A lot of the tuning is associated with the registration of the input and learned gestures. Obviously, if the registration problem can be solved, a lot of the tuning can be eliminated. It also might be the case that more sophisticated models for the human motion or a more extensive model of the human body would reduce the need for some of the parameters.

One issue that our work did not address is the differences that may occur with people tracing the same “end-effector” path with different arm and joint configurations. For example, the “bent-arm” arc used in the fourth expert-user experiment has an equivalent wrist trajectory as the “straight-arm” arc had in the first experiments. We analyzed only the wrist trajectories, although we could have additionally analyzed either the elbow positional trajectories or joint configuration trajectories. This in effect increases the size of the gesture alphabet.

We also did not address the issue of co-articulation, which involves the placement of the arm prior to and after the gesture is performed. We purposely assumed the arm movements to be independently performed and concentrated our effort on studying the dynamic movements of the arm instead of the gestural interaction aspects.

Based on our evaluation studies, we can conclude that our motion adaptation filter makes a positive contribution to the performance of gesture recognition for arm-based gestures. This seems to imply that a model of human performance can be used to eliminate some of the heuristic guess-work that must be done to make a standard gesture recognizer work. In the future, we would like to test this theory further by utilizing better models and control systems, applying the filter to other established recognition methodologies (e.g., neural network and statistical methods) and performing more extensive user studies.

## Acknowledgements

We thank Willis Marti and the Texas A&M University Computer Science Department and Visualization Laboratory for the use of computing resources. We thank Dr. Nancy Amato, Dr. Ergun Akleman and Dr. Rick Furuta for discussions contributing to the success of the work. We thank Dr. J. Edward Swan for contributions to the editing of the paper, and ITT Industries and the Naval Research Laboratory for computing support for the document preparation.

## References

- [1] R. Watson, “A Survey of Gesture Recognition Techniques”, Tech. Rep. TCD-CS-93-11, Department of Computer Science, Trinity College, Cambridge, U.K., July 1993.
- [2] J.K. Aggarwal and Q. Cai, “Human Motion Analysis: A Review”, in *IEEE Nonrigid and Articulated Motion Workshop 1997*, Piscataway, NJ, June 1997, IEEE.
- [3] V. I. Pavlovic, R. Sharma, and T. S. Huang, “Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 677–695, July 1997.
- [4] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Old Tappan, NJ: Prentice Hall PTR, 1993.
- [5] J. Martin, D. Hall, and J. L. Crowley, “Statistical Gesture Recognition through Modelling of Parameter Trajectories”, in *Gesture Workshop '99: Gesture-Based Communication in Human-Computer Interaction*, Berlin, Germany, Mar. 1999, Springer-Verlag.
- [6] G. B. Newby, “Gesture Recognition Using Statistical Similarity”, in *Virtual Reality and Persons with Disabilities Proceedings*, Northridge, CA, June 1993.
- [7] S. L. Oviatt, *Multimodal Interfaces*, pp. 286–304, Lawrence Erlbaum Assoc.:Mahwah, NJ, 2002.
- [8] P. R. Cohen, D. McGee, S. L. Oviatt, L. Wu, J. Clow, R. King, S. Julier, and L. Rosenblum, “Multimodal Interactions for 2D and 3D Environments”, *IEEE Computer Graphics and Applications*, vol. 4, pp. 10–13, July/August 1999.
- [9] A. D. Wexelblat, “A Feature-Based Approach to Continuous-Gesture Analysis”, Master’s thesis, Massachusetts Institute of Technology, May 1994.
- [10] K. V. Mardia, N. M. Ghali, T. J. Hainsworth, M. Howes, and N. Sheehy, “Techniques for On-line Gesture Recognition on Workstations”, *Image and Vision Computing*, vol. 11, no. 4, pp. 283–294, June 1993.
- [11] D. Rubine, “Specifying Gestures by Example”, in *ACM SIGGRAPH Computer Graphics Conference Proceedings*, Las Vegas, NV, Aug. 1991.
- [12] P. D. Gader, J. M. Keller, R. Krishnapuram, J.-H. Chiang, and M. A. Mohamed, “Neural and Fuzzy Methods in Handwriting Recognition”, *IEEE Computer*, vol. 30, no. 2, pp. 79–86, Feb. 1997.
- [13] K. Boehm, W. Broll, and M. Sokolewicz, “Dynamic Gesture Recognition Using Neural Networks: A Fundament for Advanced Interaction Construction”, in *SPIE Conference Electronic Imaging Science & Technology*, San Jose, CA, Feb. 1994.
- [14] L. W. Campbell, D. A. Becker, A. Azarbayejani, A. F. Bobick, and A. Pentland, “Invariant Features for 3-D Gesture Recognition”, in *Second International Workshop on Face and Gesture Recognition*, Killington, VT, Oct. 1996.
- [15] G. Herzog and K. Rohr, “Integrating Vision and Language: Towards Automatic Description of Human Movements”, in *Proceedings of the 19th Annual German Conference on Artificial Intelligence, KI-95*, Bielefeld, Germany, July 1995.
- [16] T. Starner and A. Pentland, “Real-Time American Sign Language Recognition from Video Using Hidden Markov Models”, in *Proceedings of International Symposium on Computer Vision*, Coral Gables, FL, Nov. 1995.

- [17] A. D. Wilson and A. Bobick, "Using Configuration States for the Representation and Recognition of Gesture", Tech. Rep. Technical Report No. 308, M.I.T. Media Laboratory, Cambridge, MA, June 1995.
- [18] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, New York, NY: John Wiley & Sons, Inc., 1973.
- [19] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.
- [20] C. Codella, R. Jalili, L. Koved, J. B. Lewis, D. T. Ling, J. S. Lipscomb, D. A. Rabenhorst, C. P. Wang, A. Norton, P. Sweeney, and G. Turk, "Interactive Simulation in a Multi-Person Virtual World", in *Computer Human Interaction 1992*, Monterey, CA, May 1992.
- [21] M. T. Vo and A. Waibel, "A Multi-Modal Human-Computer Interface: Combination of Gesture and Speech Recognition", in *Adjunct Proceedings of InterCHI '93*, Apr. 1993.
- [22] A. Pentland and B. Horowitz, "Recovery of Nonrigid Motion and Structure", *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 13, no. 7, pp. 730–742, 1991.
- [23] D. Metaxas and D. Terzopoulos, "Shape and Nonrigid Motion Estimation through Physics-based Synthesis", *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 15, no. 6, pp. 580–591, 1993.
- [24] A. Pentland and A. Liu, "Modeling and Prediction of Human Behavior", in *IEEE Intelligent Vehicles 1995*, Sept. 1995.
- [25] D. Metaxas, "Articulated Figure Dynamics, Behavior and Control", in *Virtual Humans: Behaviors and Physics, Acting, and Reacting (SIGGRAPH '97 Course Notes)*, Los Angeles, CA, Aug. 1997.
- [26] V. B. Zordan and J. K. Hodgins, *Tracking and Modifying Upper-Body Human Motion Data with Dynamic Simulation*, pp. 13–22, Vienna, Austria: Springer-Verlag, Sept. 1999.
- [27] N. I. Badler, Cary B. Phillips, and B. L. Webber, Eds., *Simulating Humans: Computer Graphics Animation and Control*, Oxford University Press, 1993.
- [28] C. R. Wren and A. P. Pentland, "Dynamic Models of Human Motion", in *Third IEEE International Workshop on Automatic Face and Gesture Recognition*, Nara, Japan, Apr. 1998.
- [29] C. R. Wren, B. P. Clarkson, and A. P. Pentland, "Understanding Purposeful Human Motion", in *4th International Workshop on Automatic Face and Gesture Recognition*, Grenoble, France, Mar. 2000.
- [30] K. Rohr, *Human Movement Analysis Based on Explicit Motion Models*, chapter 8, pp. 171–198, Dordrecht Boston: Kluwer Academic Publishers, 1997.
- [31] D. Hogg, "Model Based Vision: A Program to See a Walking Person", *Image and Vision Computing*, vol. 1, no. 1, pp. 5–20, Feb. 1983.
- [32] G. S. Schmidt and D. H. House, "Towards Model-Based Gesture Recognition", in *4th International Workshop on Automatic Face and Gesture Recognition*, Grenoble, France, Mar. 2000.
- [33] S. A. Su and R. Furuta, "A Logical Hand Device in Virtual Environments", in *Virtual Reality Software & Technology, Proceedings of the VRST '94 Conference*, Singapore, Aug. 1994.
- [34] D. J. Sturman, D. Zeltzer, and S. Pieper, "Hands-on Interaction With Virtual Environments", in *User Interface Software and Technology Proceedings '89*, Williamsburg, VA, Nov. 1989.
- [35] W. T. Freeman and M. Roth, "Orientation Histograms for Hand Gesture Recognition", in *IEEE International Workshop on Automatic Face and Gesture Recognition*, Zurich, Switzerland, June 1995.
- [36] K. Oka, Y. Sato, and H. Koike, "Real-time Tracking of Multiple Fingertips and Gesture Recognition for Augmented Desk Interface Systems", in *IEEE Automatic Face and Gesture Recognition*, Washington, D.C., May 2002, IEEE.
- [37] C. Lee and Y. Xu, "Online, Interactive Learning of Gestures for Human/Robot Interfaces.", in *IEEE Conference on Robotics and Automation*, Minneapolis, MN, Apr. 1996.
- [38] E. Hunter, J. Schlenzig, and R. Jain, "Posture Estimation in Reduced-Model Gesture Input Systems", in *IEEE International Workshop on Applications of Automatic Face and Gesture Recognition*, Zurich, Switzerland, June 1995.
- [39] D. J. Sturman, *Whole-hand Input*, PhD thesis, Massachusetts Institute of Technology, Feb. 1992.
- [40] J. Davis and M. Shah, "Visual Gesture Recognition", Tech. Rep. Technical Report CS-TR-93-11, Computer Vision Lab, University of Central Florida, Orlando, FL, Sept. 1993.
- [41] H. Morita, S. Hashimoto, and S. Ohteru, "A Computer Music System That Follows a Human Conductor", *IEEE Computer*, vol. 24, no. 7, pp. 44–53, July 1991.
- [42] T. Baudel and M. Beaudouin-Lafon, "CHARADE: Remote Control of Objects Using Free-Hand Gestures", *Communications of the ACM*, vol. 36, no. 7, pp. 28–35, 1993.
- [43] R. E. Kahn, M. J. Swain, P. N. Prokopowicz, and R. J. Firby, "Gesture Recognition Using the Perseus Architecture", in *Computer Vision and Pattern Recognition Proceedings*, San Francisco, CA, June 1996.
- [44] G. Welch and G. Bishop, "An Introduction to the Kalman Filter", Tech. Rep. TR 95-041, Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC, Dec. 1995.
- [45] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems", *Transactions of the ASME—Journal of Basic Engineering*, vol. 1, no. 2, pp. 34–45, 1960.
- [46] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, New York, NY: CRC Press LLC, 1993.
- [47] R. Featherstone, "The Calculation of Robot Dynamics Using Articulated-Body Inertias", *International Journal of Robotics Research*, vol. 2, no. 1, pp. 13–30, 1983.
- [48] J. J. Craig, *Introduction to Robotics: Mechanics and Control*, Reading, MA: Addison-Wesley Publishing Company, Inc., 1989.
- [49] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, New York, NY: Cambridge University Press, 1992.
- [50] S. J. Hall, *Basic Biomechanics*, St. Louis: Mosby, 1995.
- [51] I. D. Landau, *Adaptive Control, The Model Reference Approach*, New York, NY: Marcel Dekker, 1979.
- [52] D. P. Stoten and H. Benchoubane, "Empirical Studies of an MRAC Algorithm with Minimal Control Synthesis", *International Journal of Control*, vol. 51, no. 4, pp. 823–849, 1990.
- [53] G. S. Schmidt, *Model-Based Gesture Recognition*, PhD thesis, Texas A&M University, Computer Science Dept., Texas A&M University, Dec. 2000.