Motion Prediction in Gesture-based Collaborative Design Environments

Addison Chan[*] Rynson W. H. Lau[†] Lewis Li[‡]

Department of Computer Science, City University of Hong Kong, Hong Kong

Abstract

A collaborative design environment (CDE) is a shared environment over some communication networks. It allows remote users to interact with each other to perform some design tasks. To support collaborative design, the human hand is a very natural and convenient tool as it is capable of accomplishing diverse tasks such as pointing, gripping, twisting and tearing. However, there is not much work that considers using the human hand as input in CDEs, in particular over the Internet. The main reasons for this are that the Internet suffers from high network latency, which affects the interaction, and the human hand has many degrees of freedom, which adds additional challenges to synchronizing the collaboration.

In this paper, we propose a prediction method specifically designed for human hand motion to address the network latency. Through a thorough analysis of human finger motion, we have identified various finger motion constraints. By considering these motion constraints, we propose a constraint-based motion prediction method for hand motion. We present a number of experiments to demonstrate the performance of our prediction method and the dead reckoning algorithm based on the proposed predictor.

CR Categories: G3 [Probability and Statistics]: Probabilistic algorithms; I.3.2 [Computer Graphics]: Graphics Systems---Distributed / network graphics; I.3.6 [Computer Graphics]: Methodology and Techniques---Interaction techniques

Keywords: motion prediction, gesture-based application, network latency

1 Introduction

Due to globalization, more and more companies have become internationalized, with their offices scattered all over the world. During the design stage of a new product, representatives of various offices of the company may need to communicate with each other to provide suggestions on the design. They may even need to dynamically change the design together to a form acceptable to all parties.

To anticipate this change, one of our current research projects is to develop a virtual environment to support collaborative design and we call it a *collaborative design environment (CDE)*. CDE is a shared environment in which multiple users connected through the Internet may visualize and modify a design interactively. Earlier, we proposed techniques to allow multiple concurrent users to modify a design with their hands by wearing the CyberGloves [11], which are electronic gloves to capture the hand gesture. A major challenge here is to overcome the high network latency of the Internet, which has significant impact on data synchronization and user interactivity.

Because of the importance of the human hand as an intuitive interaction tool, in particular in collaborative design, we propose in this paper a prediction method specifically designed for modeling human hand motion. We have conducted a thorough analysis of human finger motion and noticed that it has an elliptic motion behavior similar to our earlier observation [4] on human hand motion in manipulating a 2D mouse. We have also identified various finger motion constraints. By considering these motion characteristics and constraints, we propose a motion prediction method for agile hand motion. Note that this work differs from existing works on hand motion capture in that we focus on modeling the state transition density rather than the posterior density of states. In addition, the intuitive constraints applied do not require extensive training. Our results show that the new method is significantly more accurate over other popular methods in predicting hand motion. Further, we show that the proposed predictor helps reduce the network loading in CDE without increasing the thresholds in dead reckoning.

The rest of the paper is organized as follows. Section 2 briefly reviews relevant work on motion prediction and human hand modeling. Section 3 presents the outline of our prediction method for hand motion. Section 4 discusses the static and the dynamic constraints of hand motion and how we consider them in our hand motion model. Section 5 demonstrates the performance of our prediction method through a number of experiments. Finally, Section 6 briefly concludes the paper.

2 Related Work

Although there has been a lot of work on motion prediction, work on human hand motion prediction is very limited. In this section, we survey different prediction methods that are developed for various applications. We then review existing work on human hand modeling and discuss the constraints in finger motion.

2.1 Motion Prediction

In distributed virtual environments, the most popular predictor used to extrapolate or predict the movement of an entity is the polynomial predictor [5]. Although polynomials of any orders may be used in the predictor, the second order polynomial predictor is commonly used. In [13], a hybrid approach is suggested. The first order polynomial is used if the acceleration is either minimal or substantial; otherwise, the second order polynomial is chosen.

There has been some work on predicting human body motion. In [1], the Kalman filter based predictor for head motion is proposed, which was originally used to filter measurement noise in linear systems by recursively minimizing the mean square estimation error. This method may work with the polynomial predictor or other prediction models to further reduce the prediction error. Experimental results show that during rapid motion, the predictor becomes less effective and its performance is similar to one without using the Kalman filter. Another method is to use the Gauss-Markov process modeling to predict human

[*] e-mail: addi@cs.cityu.edu.hk

[†] e-mail: rynson@cs.cityu.edu.hk

[‡] e-mail: kwfli@cs.cityu.edu.hk

motion [3]. Other than Kalman filtering, sequential Monte Carlo or particle filtering [7, 14] is used in motion tracking. Although this method can approximate arbitrary distributions without uni-model Gaussian assumption, it is relatively expensive computationally. [10] suggests a double exponential smoothing-based predictor, which is efficient despite of its slightly lower accuracy.

In general, specialized prediction methods usually produce more accurate results and are more capable of predicting further ahead in time. This is because they take into account the motion behavior of the target objects, which the polynomial predictor fails to do. These methods can produce reasonable results even if the prediction length is increased. A robust prediction of agile motion is thus assured. The tradeoff is the loss of generality. In this work, we propose a hand motion predictor based on an elliptical motion model to approximate the hand motion by considering the motion constraints of the human hand.

2.2 Human Hand Modeling and Motion Constraints

To model the human hand motion, we need to define a kinematical model for the hand. Some hand models have been proposed for various applications, including human hand animation and motion capture. The CyberGlove [15] uses a different model, which is similar to the one shown in Figure 1. This model resembles those used by the biomechanics specialists [2] but simplifies the complicated human hand biomechanical structure. It focuses on the major components that dominate the hand motion. Each node represents a joint and the associated arrow indicates the rotation axis. We can see that while the distal interphalangeal joints (DIJs), the proximal interphalangeal joints (PIJs) and the thumb interphalangeal joints (TIJs) have only one degree of freedom (DOF), others such as the metacarpophalangeal joints (MPJs) have two DOFs. Middle, ring and pinky have the same joint configuration as the index, although they are omitted from Figure 1 for clarity. The thumb can be considered as having similar motion as the other four fingers. However, unlike the other four fingers which motion may affect each other, the motion of the thumb is not affected by nor does it affect the motion of the other four fingers. The lines connecting joints or joints and finger tips represent bone segments. Hand motion is generated by contracting the muscles connecting adjacent bones. As muscles contract or relax, torque is produced through the tension at tendons.

Physical or kinematical constraints have been studied in modeling hand motion [12]. The most obvious one is that any rotation of a joint has a range. This range can be static or dynamic, which may be affected by other joint angles. For example, the middle finger only exhibits very limited abduction (or adduction), where abduction refers to moving two fingers away from each other and adduction refers to moving two fingers towards each other. Another example is the flexion of a finger, which is to move the finger towards or away from the center of the palm.

Dynamic constraints of finger motion may appear in many forms. For example, rotating PIJ would cause the rotation of DIJ [9]. Rotating a finger towards its palm through rotating the MPJ reduces the range of abduction of the finger [8]. As [12] points out, the flexion of a finger depends on the flexion of its neighboring fingers, i.e., the angular positions of the neighboring fingers (except the thumb) will affect the motion of a finger. In [6], principle component analysis (PCA) is used to identify the constraint, which is then used to reduce the degrees of freedom (DOFs) of the hand model. The DOFs can be further reduced by regarding the state space constructed by linear manifold spanning [16]. Both of these methods [6, 16] are generic approaches for reducing DOFs in motion tracking. However, they involve extensive training. Since the configuration spaces are different for different users or motion styles, extensive training is needed for each new user. In addition, as shown in Section 4.3.2, the correlation of the states of some fingers is non-linear. However, this is not considered by PCA. On the contrary, using explicit physical constraints does not require a computational expensive retraining process for switching users and motion styles. Only the bounds of the motion parameters are involved, which can be computed efficiently.

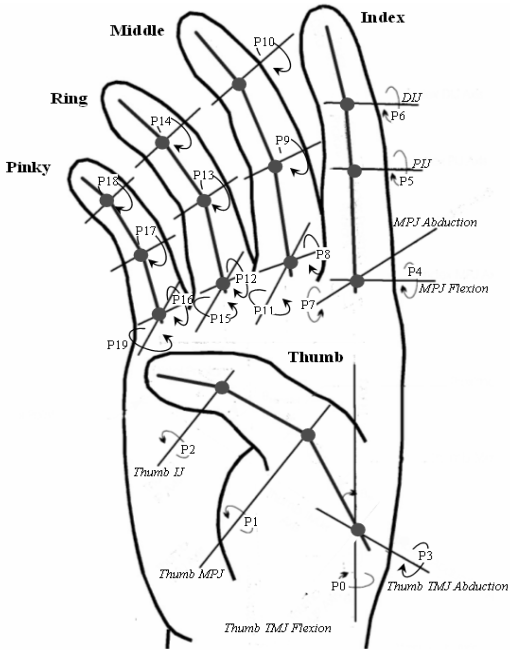


Figure 1: The human hand kinematical model.

3 A Motion Model for Human Hands

Since the motion constraints have a significant effect on finger motion, our hand motion model needs to consider these constraints in order to provide a more accurate prediction. The novelties of our work include the introduction of appropriate constraints suitable for efficient prediction without extensive training and how to integrate these constraints into a motion prediction method for hand motion prediction. Our motion model is different from those commonly used in motion tracking [6, 7, 14, 16], which are primarily based on zero-mean Gaussian or polynomial. These methods work well in tracking motion because motion tracking does not need to anticipate for long prediction length (i.e., predicting further ahead in time) or agile motion. Their focus is the posterior probability density of models' parameters. In our situation, the network latency in CDE is typically high (0.2s or above). Hence, we need to anticipate for long prediction lengths and we focus on state transition density or motion modeling [4, 13]. To improve the accuracy of the prediction, we also consider motion constraints. As will be shown later, our method is both efficient and accurate in predicting hand motion, comparing with some popular prediction methods. It also helps reduce update rates and hence network bandwidth consumption.

To develop our hand motion model, we have studied the motion data of the human hand. Figure 2 shows a typical example of the angular velocity of the index MPJ flexion. We notice that a series of pulses are generated as we move the index finger up and down.

(Similar motion patterns can also be found with other fingers.) Each of these pulses represents a single movement step of the index finger – the start of the pulse represents the start of the finger motion and the end of the pulse represents the stop of the motion. If we normalize the pulses and plot their acceleration against velocity, we may observe an elliptic pattern as shown in Figure 3, which is similar to the observation made by [4] on human hand motion in manipulating a 2D mouse. Hence, we apply the hybrid elliptic motion model suggested by [4] here for hand motion prediction. When a finger joint is at low velocity motion, we use a linear model for prediction. When its velocity picks up, we switch to an elliptic model for prediction.

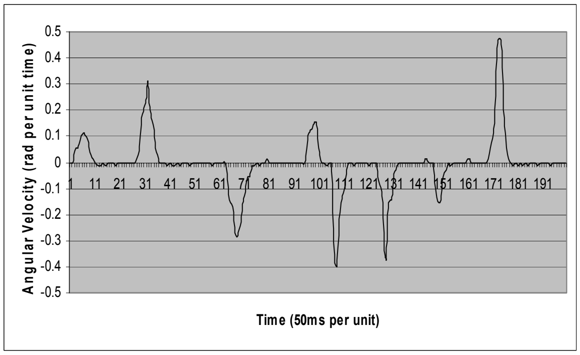


Figure 2: Movement example the index MPJ flexion.

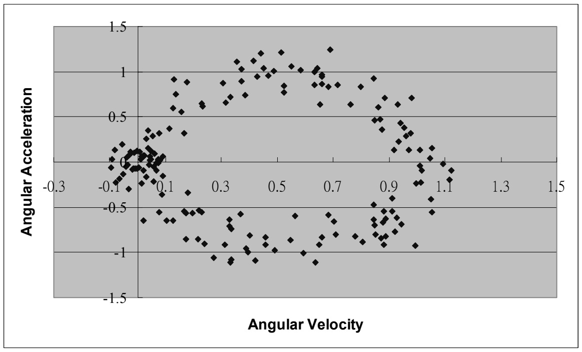


Figure 3: Motion acceleration vs. velocity.

Our method reads the state of the glove regularly, which is composed of 20 finger joint angles (the thumb: one for IJ, one for MPJ and two for TMJ; other fingers: one for DIJ, one for PIJ and two for MPJ) as shown in Figure 1. 20 Kalman filters are used to filter the measurement noise of the 20 input joint angles. We input the joint angles read from the glove to the predictors. Each predictor will then predict the joint angle at the end point of the corresponding pulse. We use the static constraints and the dynamic constraints to verify the predicted joint angles. If a predicted joint angle violates some constraints, we may need to adjust the parameters (i.e., the state vectors) of the relevant predictors in order to revise the predicted joint angle.

We consider the static constraints of each joint angle first. If the predicted joint angle at the end of the pulse exceeds its static limits, the state vector of the corresponding predictor is adjusted. Second, we consider the dynamic constraints of the joint angles, which always involve the motion of two or more neighboring fingers. The MPJ flexions of the index, middle, ring and pinky are considered. (Since the motion of the thumb does not affect the motion of other fingers, it needs not be considered here.) We compare the predicted angular differences of neighboring MPJ flexion angles (index-middle, middle-ring and ring-pinky) at the end of their pulses. If any pair of the fingers exceeds its corresponding dynamic limits, the state vectors of the two predictors are adjusted. The dynamic constraints between the MPJ flexion angles and the MPJ abduction angles of neighboring fingers are then considered. Usually, flexion involves more energetic motion than abduction and we adjust the state vectors of the predictors for abduction according to the predicted angles at the end of the MPJ flexion pulses. However, if abduction is found to be more energetic, the state vectors of the predictors for the MPJ flexion will be adjusted instead.

After the above adjustments, the estimated end point of each joint angle will satisfy the constraints for the current moment. We may now use the predictors to predict the future angles of individual finger joints. The whole prediction method repeats when a new set of joint angles is read from the glove. Since the constraints consist of upper and lower bounds of motion only, we may simply adjust the parameters of the constraints dynamically by recording the actual maximum and minimum values of the motion.

4 FINGER MOTION FORMULATIONS

In this section, we first summarize our motion prediction method for human hands. We then discuss how to determine the state vector for the Kalman filter and how to adjust the state vector to satisfy the static constraints. Finally, we discuss the dynamic constraints existed between neighboring fingers and how to further adjust the state vector to satisfy the dynamic constraints. We apply these two types of motion constraints, which are not acquired from training, to our prediction.

4.1 The Hybrid Motion Prediction Method

In Figures 2 and 3, the angular velocity, $\omega_t(X)$, within a pulse can be approximated by an *elliptic model*, given the state vector X=(K_1, K_2):

$$\omega_t(X) = K_1(\cos(K_2\ t) - 1) \qquad (1)$$

where t is the time relative to the beginning of the pulse. X is a non-zero constant vector. To reduce the measurement noise and the error due to the elliptic approximation, we apply an extended Kalman filter with hybrid process noise covariance matrix on X, computed from Eq. (1), of each joint. Hence, X is considered as the components of the *state vector* to the Kalman filter. The detail of this can be found in [4]. Hence, the predicted angle of a joint can be found by integrating $\omega_t(X)$ in Eq. (1) along the predicting interval and is calculated as:

$$\theta_{pred}(t_{pred}) = \theta_{cur} + [(K_1\sin(K_2\ t)/K_2) - K_1 t]_{t_{cur}}^{t_{pred}} \qquad (2)$$

where t_{cur} is the current time. θ_{pred} and θ_{cur} are the predicted and the current joint angles, respectively. t_{pred} is the prediction time. If we let τ be the prediction length, t_{pred} can be calculated as: $t_{pred} = min(t_{cur}+\tau, 2\pi/K_2)$.

From Figure 2, we observe that the angular velocity is nearly zero during the period between any two pulses. It is not appropriate to model the velocity using the elliptic model due to the increased significance of noise and distortion on the sampled data. Hence, we model low velocity motion with a generic linear model commonly used in dead reckoning. Note that while a joint angle is approximated by a linear model, the Kalman filter is not used. Once the angular velocity is high enough, we switch to use the elliptic model. However, to do this, we need to determine the values of the state vector X. This will require three data point pairs, (t, ω), to be sampled and neither of them should be (0, 0). Hence, when a new pulse is detected, the first three velocity values which are higher than a threshold value are used to determine the state vector.

4.2 Determining the State Vectors

To apply the Kalman filter to filter sampling noise and the error due to the elliptic approximation, X can be regarded as the components of the state vector to the Kalman filter. The state transition matrix ϕ_k, which relates the states at time t_k and t_{k+1}, is set to be I because the state vectors remain unchanged within one pulse. The measurement matrix H_k is defined as the gradient of $\omega_t(X)$, which is $(\cos(K_2\ t)\text{-}1,\ \text{-}K_1\ t\ \sin(K_2\ t))^T$. The process noise and measurement error covariance matrix can be tuned by the Powell's method.

This approach is different from [1] since our method does not take the velocity and acceleration values as the components of the state vector. Instead, we first determine the preliminary state vector of the Kalman filter. The initial guess of the state vector at the start of each pulse is crucial for the convergence of the Kalman filter as well as the accuracy of the prediction. We use three velocity values to compute the vector and need to find the optimized X that matches the three velocity values. Let $W = (\omega_1, \omega_2, \omega_3)^\mathrm{T}$, consisting of the first, second and third angular velocity values, respectively, taken since the start of a pulse. We define the optimization problem as to minimize $f(X)$:

$$f(X) = \| g(X) - W \|^2 \qquad (3)$$

where $g(X) = (\omega_1(X), \omega_2(X), \omega_3(X))^T$. To speed up the optimization process and to guarantee a global minimizer, an explicit solution is beneficial. The solution to the problem can be found by setting the gradient of $f(X)$ to zero. This can then be solved by the Newton's method or other numerical methods. Note that if we use more velocity values in the initial guess of the state vector, the estimation will be more accurate. However, the computation will become more complex and thus delay the output of the estimation.

Each joint angle should have lower and upper flexion limits, which are independent of other joints. They are referred as *static constraints* [12]. If the state vector $(K_1, K_2)^\mathrm{T}$ estimated by Eq. (3) causes violation of these constraints, it needs to be evaluated again by another formula. The angle of a joint, θ, should satisfy the following inequalities:

$$smin \le \theta \le smax \qquad (4)$$

where *smin* and *smax* are the lower and the upper limits. Each joint has different values of *smin* and *smax*. Let ψ be the initial angular position at the start of the pulse. After the preliminary state vector $(K_1, K_2)^\mathrm{T}$ is found, we predict the value of θ at the end of the pulse by Eq. (2) as follows:

$$\theta = \psi + (\text{-}2\pi K_1 / K_2) \qquad (5)$$

If the constraint in Eq. (4) is violated, it can be rewritten as:

$$\varphi = -2\pi K_1 / K_2 \qquad (6)$$

where $\varphi = smax - \psi$ or $smin - \psi$. If we substitute Eq. (6) into Eq. (3), a similar optimization problem but with only one variable can be obtained. Formally, the solution to the problem can be obtained by minimizing $f(X)$ in Eq. (3) subject to the linear constraint in Eq. (6). Again, the solution can be found by setting the gradient of $f(X)$ to zero.

4.3 Dynamic Constraints of Finger Motion

When two or more neighboring fingers are moving, some constraints may exist among these fingers. These constraints vary as the fingers move and are referred to as *dynamics constraints*. To avoid overcomplicating the hand model, we only focus on two important types of dynamic constraints in our method. One exists in MPJ flexion and the other exists between MPJ flexion and MPJ abduction. As mentioned before, the motion of the thumb is independent of other fingers and so we only need to consider the dynamic constraints among the other four fingers here. The bounds defined by the constraints can be easily obtained by recording the maximum absolute values caused by the motion.

4.3.1 Dynamic Constraints of MPJ Flexion

The dynamic constraints of MPJ flexions specify the angular difference between any two neighboring fingers as follows:

$$F_{min} \le D\, X_{flex} \le F_{max} \qquad (7)$$

where $$D = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \\ -1 & 0 & 0 & 1 \end{pmatrix} \qquad (8)$$

X_{flex} is a vector composed of the angular positions of the index MPJ flexion to the ring MPJ flexion as $(\theta_{ind\_mpj\_flex,}\ \theta_{mid\_mpj\ flex,}\ \theta_{rin\_mpj\_flex,}\ \theta_{pin\_mpj\_flex})^\mathrm{T}$ and *ind*, *mid*, *rin* and *pin* refer to the index, middle, ring and pinky, respectively. F_{max} and F_{min} are vectors which components denoting the relative minimum and maximum differences among the four MPJ flexions, i.e., $\theta_{ind\_mpj\_flex}–\theta_{mid\_mpj\ flex}$, $\theta_{mid\_mpj\ flex}–\theta_{rin\_mpj\_flex}$, $\theta_{rin\_mpj\_flex}–\theta_{pin\_mpj\_flex}$ and $\theta_{pin\_mpj\_flex}–\theta_{ind\_mpj\_flex}$. We calculate the expected angular position of each joint at the end of the motion pulse with Eq. (6). The angular positions are then used in Eq. (7) to check for the violation of dynamic constraints. If any of the constraints is violated, the state vector of the corresponding flexion may need to be adjusted.

In order to apply the constraints defined by Eq. (7), we need to study how the pulse length of an MPJ flexion may be affected by other MPJ flexions. We conduct an experiment by moving the fingers arbitrarily and examine how much the pulse length of a finger is altered by the movement of other fingers. We have obtained thousands of samples from several users. The results as shown in Table 1 indicate that the pulse length of the flexion motion is not really affected by the movement of other fingers. Again, the MPJ flexions of index to pinky are examined. We can predict the pulse length by using the formula, $2\pi / K_2$. The absolute difference between the actual and the predicted pulse lengths is defined as the error value. The ratio of the average error to the average pulse length is about 0.1 which is a small value.

Table 1: The change in pulse length due to the motion of other fingers.

Average pulse length	491 ms.
Average error in predicted pulse length	47 ms.
Average error / average pulse length	0.096

As two neighboring fingers move in the opposite directions, their moving speeds are lowered by each others if they are close to their corresponding dynamic limits. It would make the pulse lengths longer. On the other hand, the room for moving is reduced, which would shorten the pulses. These contradicting factors appear to cancel each another. Besides, we have attempted to move the MPJs of two neighboring fingers and stop at their corresponding dynamic limits. The fingers from index to pinky are examined. Table 2 is then drawn. The average length of all motion pulses and the average absolute differences in stopping time between two correlating flexions are recorded. We have found that the ratio of average difference to the average pulse length is small enough to conclude that the MPJ flexions of two correlating fingers stop simultaneously if they reach their corresponding dynamic constraints.

Table 2: Simultaneous stopping of two neighboring MPJ flexions.

Average pulse length	449 ms.
Average stopping difference	12 ms.
Average difference / average pulse length	0.027

One can simply adjust MPJ flexion states, X, by minimizing $\sum_{X \subset S_{flex}} f(X)$ subject to the linear constraints in Eq. (7), where S_{flex} is the set of MPJ flexion states. The dimension of the problem is 8. This is similar to the optimization problem in inverse kinematics. As it has a high computational cost, it is not suitable for real-time prediction. From the results shown in Tables 1 and 2, it is possible to reduce the cost by half. Since we have assumed that the pulse length remains the same, K_2 should not change. However, we note from Table 2 that there is a small change in the pulse length. Instead of adjusting K_2, we add a δ to K_2 and then minimize $\| \delta \|^2$ subject to:

$$2\pi / (K_2^{(i)} + \delta^{(i)}) - t_{cur}^{(i)} = 2\pi / (K_2^{(j)} + \delta^{(j)}) - t_{cur}^{(j)} \qquad (9)$$

where $\delta = (\delta^{(i)}, \ \delta^{(j)})^T$. i and j are the MPJ flexions of the two neighboring fingers, which predicted positions violate the constraints defined by Eq. (7). t_{cur} is the time elapsed since the start of the pulses. The R.H.S and L.H.S in Eq. (9) represent the time left before both fingers are stopped by their corresponding MPJ flexion dynamic constraints. They are expected to stop at the same time. K_2 is then incremented by δ. The complexity of the problem can be further reduced by rearranging and trimming the constraints stated in Eq. (7) and (8). After the state vectors of each MPJ flexion joint have been adjusted by considering the static constraints, the angular positions in Eq. (5) are checked against the inequality in Eq. (8). The columns and rows of D corresponding to the joints not violating the constraints are eliminated. K_2 of some joints are removed from the problem because they are adjusted by the method described above. As a result, the size of matrix D is decreased. The constraint in Eq. (7) is thus changed to:

$$D^* X_{flex}^* \leq F^* \qquad (10)$$

where the components of X_{flex}^* are picked from X_{flex} and the size of X_{flex}^* is smaller than X_{flex}. Since only minimum or maximum bound can be reached at one time, F_{min} and F_{max} can be combined into F^* by negating appropriate rows in D.

Although the complexity of the problem is highly reduced, we do not apply Eq. (10) to minimize $\sum_{X \subset S_{flex}} f(X)$ because the characteristics of the finger motions will be ignored. We have observed that when the fingers of a free hand are moving vigorously, some of the fingers may violate their dynamic constraints. If a pair of fingers reach the limits first, they may affect the motion limits (maximum or minimum) of other fingers. This is not considered by Eq. (10). The pair of joints which reach their limits first will be handled first. We compute the remaining time or length of the pulse l_{rem}, which is equal to $2\pi / K_2 - t_{cur}$. K_2 has already been adjusted as described above. We choose the MPJ flexions of two neighboring fingers, i and j, violating the dynamic constraints and with at least l_{rem}. Since K_2 of both flexions have been adjusted, we only need to minimize the following functions:

$$f_{K1}^{((i,t_i),(j,t_j))}(V_{K1}) = \sum_{n=i,j} \| g_{K1}^{(n,t_n)}(K_1^{(n)}) - W^{(n,t_n)} \|^2 \qquad (11)$$

where $g_{K1}^{(n,t_n)}(K_1^{(n)}) = (\omega_{tn+1}(X(K_1^{(n)})), \ \omega_{tn+2}(X(K_1^{(n)})), \ \omega_{tn+3}(X(K_1^{(n)})))^T$, $W^{(n,t_n)} = (\omega_{tn+1}, \omega_{tn+2}, \omega_{tn+3})^T$, t_n is the time elapsed since the start of the motion pulse of joint n, and $V_{K1} = (K_1^{(i)}, K_1^{(j)})^T$. $X(K_1^{(n)})$ is the state vector with a fixed $K_2^{(n)}$. The above minimization is subject to $\theta^{(i)}_{pred}(2\pi/K_2^{(j)}) - \theta^{(j)}_{pred}(2\pi/K_2^{(j)}) = F^*_{i\text{-}j}$, where $F^*_{i\text{-}j}$ is one of the components in F^* which correlates MPJ flexions i and j. The function $\theta^{(i)}_{pred}(2\pi/K_2^{(j)})$ and $\theta^{(j)}_{pred}(2\pi/K_2^{(j)})$ are defined by (2). The constraint is actually extracted from the one in Eq. (10). This can be easily transformed into a linear form as follows:

$$A^T V_{K1} = b \qquad (12)$$

where A and b are some constant vector and scalar, respectively.

After the first pair of state vectors have been determined, we then adjust other joints which violate the flexion constraints. Based on the adjusted pair of joints, we may check the constraints again and iterate the process just depicted. The pair of joints with the least l_{rem} are sought and adjusted. As the iteration continues, more state vectors are fixed. This reduces the computational cost of the minimization problem. The iteration stops when there are no more flexion constraints violated. We would like to point out here that once all violated constraints of the moving fingers are adjusted, there is no need to check for violation until another finger suddenly starts to move. Only then, we will check for constraint violation again.

4.3.2 Dynamic Constraints between MPJ Flexion and Abduction

After adjusting the MPJ flexions, the MPJ abduction states may also need to be adjusted according to the dynamics constraints existed between flexion and abduction. This may in turn require the adjusted MPJ flexions to be modified again. As mentioned in Section 2, the abduction limits of each finger depend on the MPJ flexion of the same finger. To study the relationship, we have captured and plotted the index MPJ abduction against the index MPJ flexion as shown in Figure 4. The data is taken by stretching the index MPJ in all direction so as to draw a circle as large as possible. This helps find the dynamic limits of abduction at every MPJ flexion angle. As shown in Figure 3, the range of abduction angle reduces as the MPJ flexion approaches its static limits and increases as the MPJ flexion approaches the midpoint between the static limits. We may approximate the data points by two parabolas or quadratic curves, one above and the other below the x-axis. The widths of the two parabolas indicate the static limits of the MPJ flexion, while the heights indicate the static limits of the MPJ abduction. Similar graphs can be found by moving the MPJ of other fingers.

Here, we attempt to relate the dynamic abduction limits, d_{min_abd} and d_{max_abd}, with the MPJ flexion angle, θ_{flex}, by a quadratic model. To model the dynamic abduction maximum, i.e., the upper curve in Figure 4, we need to obtain three points to define the curve. We observe that the curve cuts the x-axis at two points, the static minimum (s_{min_flex}, 0) and static maximum (s_{max_flex}, 0) of the MPJ flexion. If we also consider the midpoint between them, i.e., $((s_{min\_flex}+s_{max\_flex})/2, \ s_{max\_abd})$, then d_{max_abd} can be computed as:

$$d_{max\_abd} = -4 s_{max\_abd} / (s_{min\_flex} - s_{max\_flex})^2 \, (\theta_{flex}^2 - (s_{min\_flex} + s_{max\_flex}) \, \theta_{flex} + s_{min\_flex} \, s_{max\_flex}) \qquad (13)$$

where $s_{min\_flex}/s_{max\_flex}$ are the static limits of the MPJ flexion. Similarly, the lower curve can be defined by three points: (s_{min_flex}, 0), (s_{max_flex}, 0), and $((s_{min\_flex}+s_{max\_flex})/2, \ s_{min\_abd})$, where s_{min_abd}/s_{max_abd} are the static limits of the MPJ abduction. Hence, d_{min_abd} can be computed as:

$$d_{min\_abd} = -4 \, s_{min\_abd} / (s_{min\_flex} - s_{max\_flex})^2 \, (\theta_{flex}^2 - (s_{min\_flex} + s_{max\_flex}) \, \theta_{flex} + s_{min\_flex} \, s_{max\_flex}) \qquad (14)$$

Note that MPJ flexion usually possesses greater motion and is likely more energetic than MPJ abduction to dominate the MPJ joint motion. Energetic motion usually produces greater velocity. This reduces the significance of measurement errors. Hence, we may assume MPJ flexion, rather than MPJ abduction, to be the dominating factor, as in Eq. (13) and (14).

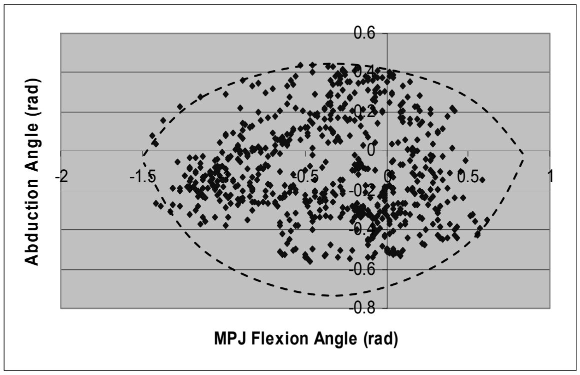


Figure 4: Index MPJ abduction against index MPJ flexion.

After the state vector for the MPJ flexion has been evaluated by the method shown in Section 4.3.1, the dynamic abduction limits of each finger are found by Eq. (13) and (14). We use θ_{flex} at the end of the velocity pulse, which is evaluated by Eq. (5), to estimate the values of d_{min_abd} and d_{max_abd}. The predicted abduction angle θ_{abd} at the end of the pulse, which is again evaluated by Eq. (5), is then checked against the following constraints:

$$\theta_{abd} \geq d_{min\_abd} \tag{15}$$

$$\theta_{abd} \leq d_{max\_abd} \tag{16}$$

Violation of any one of the above constraints requires the adjustment of the state vector $(K_1{}^{abd}, K_2{}^{abd})^{\mathrm{T}}$ for the MPJ abduction. Since Eq. (15) and (16) are a complimentary pair, it is not possible to violate both. If Eq. (15) is violated, the constraints can be satisfied by simply multiplying K_1 with a factor. Thus, we adjust the state vector in the following way:

$$K_1{}^{abd} = (\Omega / \Omega_{ori})K_1{}^{abd\_ori} \tag{17}$$

Ω and Ω_{ori} are found as:

$$\Omega = d_{min\_abd} - \theta_{abd\_cur} \tag{18}$$

$$\Omega_{ori} = -2\pi K_1{}^{abd\_ori} / K_2{}^{abd\_ori} - (K_1{}^{abd\_ori} / K_2{}^{abd\_ori} \sin(K_2{}^{abd\_ori}\, t_{cur}) - K_1{}^{abd\_ori}\, t_{cur}) \tag{19}$$

where $(K_1{}^{abd\_ori}, K_2{}^{abd\_ori})^{\mathrm{T}}$ is the state vector before the adjustment and θ_{abd_cur} is the current abduction angle. Here, $K_2{}^{abd}$ needs not be adjusted as we assume that the pulse duration is not changed (Section 4.3.1 and Table 1). On the other hand, if Eq. (16) is violated, Eq. (18) is replaced by:

$$\Omega = d_{max\_abd} - \theta_{abd\_cur} \tag{20}$$

Sometimes, abduction may be energetic enough to alter flexion, as flexion is stationary or has a weak motion, i.e., flexion is approximated by the linear model. In this case, if the predicted θ_{abd} at the end of the pulse violates either Eq. (15) or (16), a new θ_{flex} is evaluated by solving Eq. (13) or (14), where d_{min_abd} and d_{max_abd} are known values. Two solutions are usually found and we choose the one nearer to the current flexion angle θ_{flex_cur}. The angular velocity of flexion ω_{flex} is set as:

$$\omega_{flex} = (\theta_{flex} - \theta_{flex\_cur}) / l_{rem} \tag{21}$$

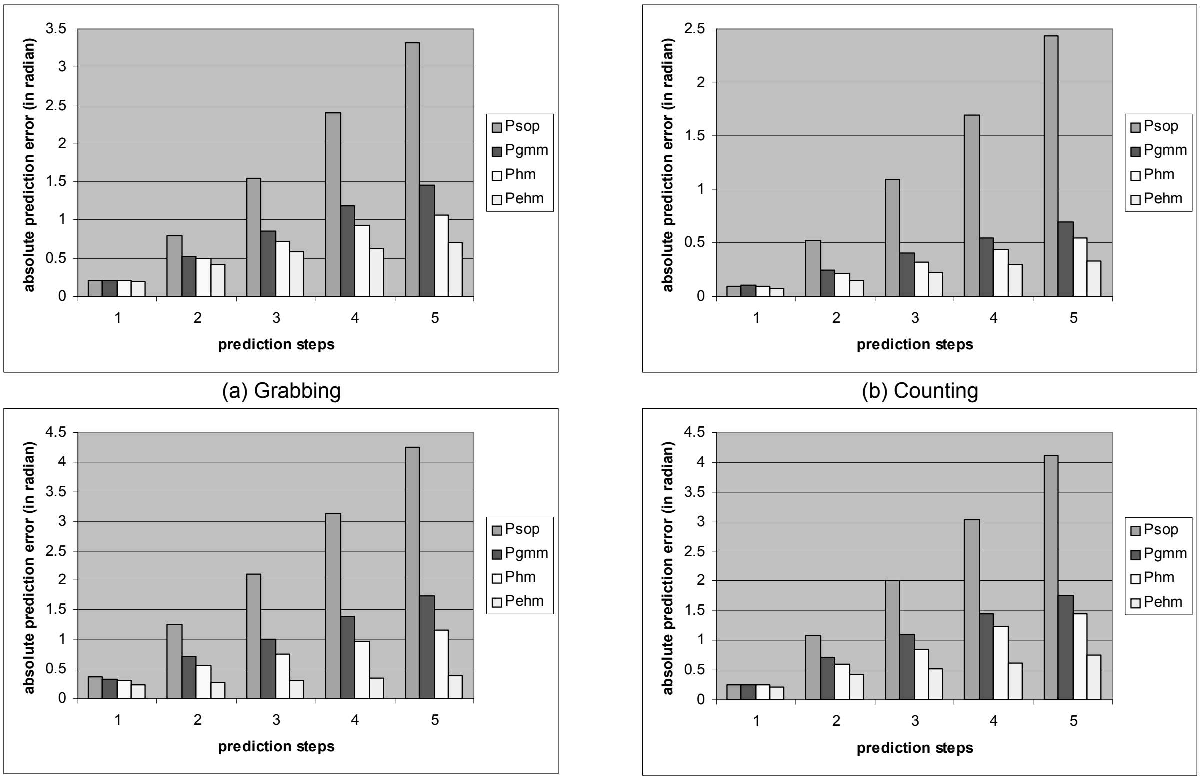
where l_{rem} is the remaining time or pulse length of the velocity pulse for abduction, as mentioned in the previous section.

5 Results and Discussions

To study the performance of the proposed prediction method, we have implemented and compared the performance of four predictors:

- The second order polynomial predictor [1]: P_{sop}
- The Gauss-Markov model [3]: P_{gmm}
- The hybrid motion model without considering any physical constraints [4]: P_{hm}
- The new constraint-based hybrid motion model: P_{ehm}

P_{sop} and P_{gmm} are defined as follows:



(a) Grabbing

(b) Counting

(c) Typing

(d) Sculpting

Figure 5: Comparison of prediction errors of the four prediction methods with four types of hand action.

$$\boldsymbol{P_{sop}}: \quad \theta_{pred} = \theta_{cur} + \omega\, t + 0.5\alpha\, t^2 \tag{22}$$

$$\boldsymbol{P_{gmm}}: \quad \theta_{pred} = \theta_{cur} + \omega\, t + \alpha/\beta\, (t + e^{-\beta t}/\beta - 1/\beta) \tag{23}$$

where ω is the angular velocity, α is the angular acceleration, t is the time elapsed, and β is a predefined parameter of the Gauss-Markov model. All these methods use Kalman filtering and the parameters are found by the Powell's method. We have integrated these methods into a glove-based system for collaborative design [11]. Besides, we attempt to visualize the relationship between the deviations of the predicted states of each predictor and the corresponding network loadings by varying the threshold value. The accuracy of each predictor at the same network loading level can be compared. The comparison of their computation costs is omitted since we found that all predictors spend only few milliseconds in computing one prediction.

5.1 Prediction Errors

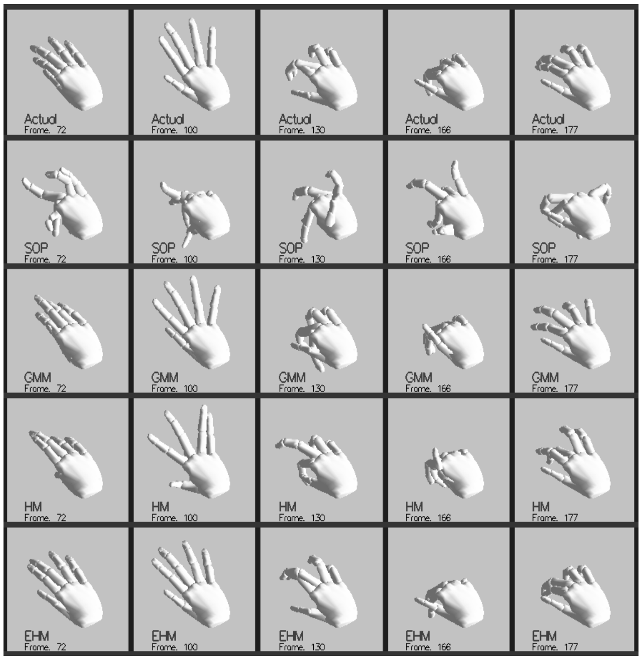


Figure 6: Performance comparison on predicting sculpting gestures at 0.5s intervals: (a) actual, (b) P_{sop}, (c) P_{gmm}, (d) P_{hm}, and (e) P_{ehm}.

Four types of typical hand actions are used in our experiments: *grabbing*, *counting*, *typing* and *sculpting*. Sculpting involves a significant amount of random finger motion that includes flexion as well as abduction. It may involve the characteristics of the other three types of hand actions. The motions from ten users are captured. Both experts and novices are involved.

Figure 5 shows the prediction errors when applying the four prediction methods on the four types of hand actions. Each *prediction step* represents a 0.1s time interval. The prediction error E (the vertical axis) is measured as the average difference between the actual and the predicted joint angles in radian as:

$$E = \sum_{t=1}^{N}\sum_{i=1}^{I}\frac{|\theta_{i,t} - \hat{\theta}_{i,t}|}{N * I} \tag{24}$$

where i is the index to a joint angle, t is the sample number, I is the total number of joints, N is the total number of samples, $\theta_{i,t}$ is the sampled angle, and $\hat{\theta}_{i,t}$ is the predicted angle.

In general, the prediction errors of the four prediction methods increase roughly linear with the increase in *prediction length*, which specifies how far ahead in time (in terms of prediction steps) that we want to predict. However, each method increases at a different rate. At low prediction length (0.1s), the difference in performance of all four methods is not so obvious, although we may observe that P_{ehm} (the proposed method) does slightly better than other methods. However, as the prediction length increases, P_{ehm} clearly outperforms all the other methods and P_{sop} performs the worst.

Consider Eq. (22) for P_{sop} and Eq. (23) for P_{gmm}. If we differentiate the two equations, we get:

$$\boldsymbol{P_{sop}}: \quad \omega_{pred} = \omega_{cur} + \alpha\, t \tag{25}$$

$$\boldsymbol{P_{gmm}}: \quad \omega_{pred} = \omega_{cur} + \alpha\, (1 - e^{-\beta t}) / \beta \tag{26}$$

If we compare Eq. (1), (25) and (26), we can see that all three equations are increasing function when t is small. However, as t increases, P_{sop} will continue to increase, P_{gmm} will gradually level off, while the elliptic model of P_{hm} and P_{ehm} will peak at some point and then gradually return to zero. This explains why the prediction errors of the four methods are very close at low prediction length. As the prediction length increases, the velocity curve of P_{sop} deviates more and more from the shape of the pulses in Figure 2, while the predicted velocity curves of P_{hm} and P_{ehm} resemble more closely to the shape of the pulses. This explains that both P_{hm} and P_{ehm} perform better than the other two methods. On the other hand, P_{ehm} considers the static and dynamic motion constraints and uses them to adjust the state vectors for prediction. This allows P_{ehm} to have a much lower prediction error than P_{hm} at high prediction length.

Comparing the four types of action, we may notice that the prediction error for counting in Figure 5(b) is generally lower than those for other actions. As mentioned earlier, counting involves the rapid movement of only one or a few joints. Most of the other joints are nearly stationary. Hence, only a few MPJ flexions contribute to the prediction error. This is why counting produces a lower prediction error compared with other actions. Grabbing is similar to counting but with more joints moving at the same time. Thus, the prediction error of grabbing is higher than that that of counting. Although typing also involves the rapid movement of only a few joints at any time, the movement usually lasts for longer period of time. Sculpting, on the other hand, also involves a longer period of finger movement. This is why the prediction errors for typing and for sculpting are relatively high compared with the other two actions.

As shown in Figure 5, it is obviously that our proposed predictor outperforms others if the prediction length is greater than one step. Figure 6 shows some snapshots of the predicted hand gestures using different prediction methods with a rather long prediction length of 0.5 second.

5.2 Dead Reckoning Performance Evaluation

To study the relationship between prediction accuracy and network loading, we plot the difference between the real and the predicted states (average error in radian) against the network loading with different network latencies. The results are showed in Figure 7. The network loading is measured as the average number of 32-bit floating point datum sent through the network to update the states of a remote machine per sampling step (0.05s). Like Figure 5, the average error here is found by Eq. (24). The sample points in Figure 7 are obtained by varying the threshold value from 4 radians to 12 radians. This threshold value is the sum of all the absolute differences between the real and the predicted joint angles. The users move their hands arbitrarily throughout the experiments.

Figure 7 shows that the average error increases with the network latency, which agrees with the results shown in Figure 5.

With higher network latency, the predictors will have to predict further ahead in time, which increases the average error. We can see that the performance of P_{ehm} is much better than others as the network latency increases, which again agrees with the results shown in Figure 5. This will lead to a better performance in dead reckoning. Note that the prediction error of P_{sop} is too high in Figure 7(c) that its curve is not shown.

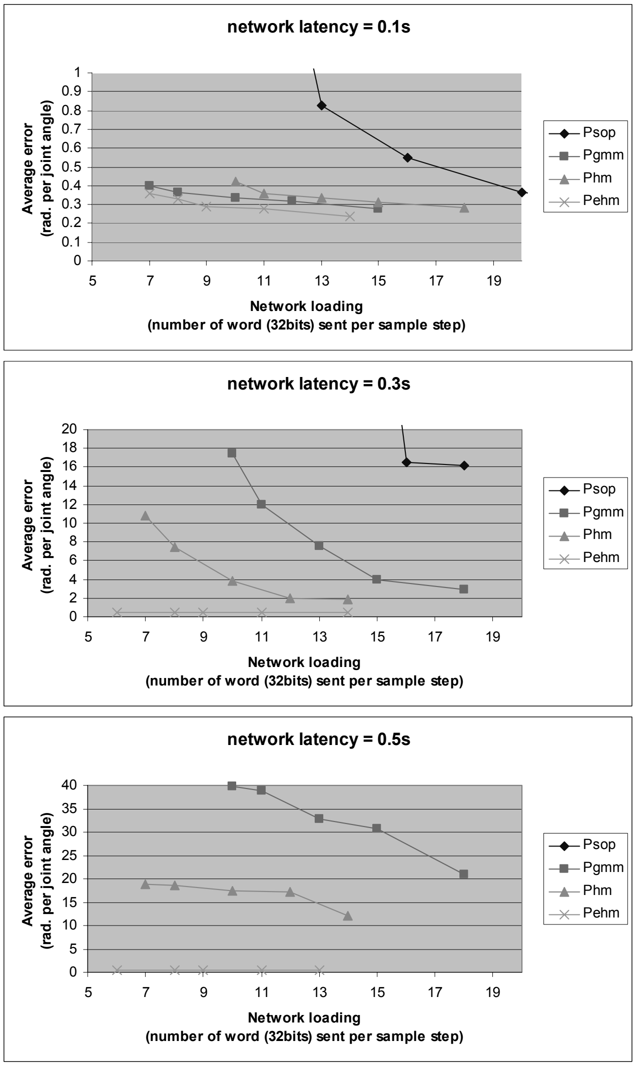


Figure 7: Performance comparison of the four predictors in dead reckoning, at different network latency: (a) 0.1s, (b) 0.3s, and (c) 0.5s.

6 Conclusions

In this paper, we have presented a motion prediction method for human hands by considering the physical constraints of finger motion. We classify the constraints into static and dynamic. By predicting if a moving joint will violate the constraints at the end of its movement step, we may adjust the state vector of its predictor. This provides a feedback to the predictor to improve its accuracy. We have demonstrated with some experiments that the proposed predictor performs much better than other popular prediction methods, in particular in long prediction length. The proposed predictor also performs better when it is applied to dead reckoning. The new method is expected to be useful in applications that require the remote sharing of hand gesture, such as collaborative design.

Acknowledgements

We would like to thank Frederick Ng for helping in some earlier implementation work. The work described in this paper was partially supported by a CERG grant from the Research Grants Council of Hong Kong (RGC Reference: CityU 1133/04E) and a SRG grant from the City University of Hong Kong (Project Number: 7001465).

References

[1] R. Azuma and G. Bishop. “A Frequency-Domain Analysis of Head-Motion Prediction.” *Proc. ACM SIGGRAPH'95*, pp. 401-408, 1995.

[2] E. Biryukova and V. Yourovskaya. “A Model of Human Hand Dynamics.” *Advances in the Biomechanics of the Hand and Wrist*, Plenum Press, pp. 107-122, 1994.

[3] T. Capin, I. Pandzic, N. Magnenat-Thalmann, and D. Thalmann. “A Dead-Reackoning Algorithm for Virtual Human Figures.” *Proc. IEEE VRAIS’97*, pp. 161-169, 1997.

[4] A. Chan, R.W.H. Lau, and B. Ng. “Motion Prediction for Caching and Prefetching in Mouse-Driven DVE Navigation.” *ACM Trans. on Internet Technology*, **5**(1):1-22, 2005.

[5] DIS Steering Committee. “IEEE Standard for Distributed Interactive Simulation - Application Protocols.” *IEEE Standard 1278*, 1998.

[6] T. Heap and D. Hogg. “Towards 3D Hand Tracking using a Deformable Model.” *Proc. IEEE Int’l Conf. on Automatic Face and Gesture Recognition’96*, pp. 140-145, 1996.

[7] M. Isard and A.Blake. “CONSENSATION – Cconditional Density Propagation for Visual Tracking.” *Int’l Journal of Computer Vision*, **29**(1):5-28, 1998.

[8] I. Kapandji. *The Physiology of the Joints (Vol. 1)*, Churchill Livingstone, 1982.

[9] J. Landsmeer. “The Coordination of Finger Joint Motions.” *Journal of Bone and Joint Surgery*, **45**:1654-1662, 1963.

[10] J. LaViola Jr. “An Experimental Comparing Double Exponential Smoothing and Kalman Filter-Based Predictive Tracking Algorithms.” *Proc. IEEE VR’03*, pp.283-284, 2003.

[11] F. Li, R.W.H. Lau, and F. Ng. “Collaborative Distributed Virtual Sculpting.” *Proc. IEEE VR’01*, pp. 217-224, 2001.

[12] J. Lee and T. Kunii. “Model-Based Analysis of Hand Posture.” *IEEE Computer Graphics and Applications*, **15**(5):77-86, 1995.

[13] S. Singhal and D. Cheriton. “Exploiting Position History for Efficient Remote Rendering in Networked Virtual Reality.” *Presence*, **4**(2):169-193, 1995.

[14] B. Stenger, A. Thayananthan, P. Torr, and R.Cipolla. “Filtering Using a Tree-Based Estimator.” *Proc. IEEE ICCV’03*, pp. 1063-1070, 2003.

[15] Virtual Technologies. *VirtualHand Software Library Reference Manual*, Virtual Technologies, 1998.

[16] H. Zhou and T. Huang. “Tracking Articulated Hand Motion with Eigen Dynamics Analysis.” *Proc. IEEE ICCV’01*, pp. 1102-1109, 2003.