Motion Data Alignment For Real-Time Guidance in Avatar Based Physical Therapy Training System

Dennis Shen, *Member*, *IEEE*, Yao Lu, *Member*, *IEEE*, and Sujit Dey, *Fellow*, *IEEE*

***Abstract*— In this paper, we propose an Avatar based Virtual Reality user training system that efficiently trains users in performing a variety of activities using a pre-recorded avatar. To evaluate and monitor the user's adherence to the avatar's instructions, the system compares the user's motion data against the avatar's motion data, with the latter established as the ground truth dataset. Unfortunately, human reaction delay may cause the motion sequences between the user and the avatar to be misaligned. Consequently, to enable accurate comparison, we analyze four signal processing time delay estimation methods—an existing method and three proposed methods—to align the motion sequences between the user and pre-recorded avatar, allowing the correct frames to be compared. Our experiments demonstrate that the proposed methods perform better data alignment than the existing method and the fourth method, which employs a novel spatial-temporal segmentation algorithm, has the highest potential to be the optimal delay estimation approach. Further, to provide real-time guidance to the user, we determine a unique tolerance threshold for each activity such that a user accuracy value below the threshold value prompts real-time guidance to correct the user and an accuracy value above the threshold is tolerated. We perform an experiment with the assistance of a physical trainer and use the experimental data to design a histogram-based method using Bayesian decision theory to determine the threshold values.**

Keywords—Virtual Reality, signal processing, gesture recognition, pattern classification

I. Introduction

Often times, after a physical therapy or fitness training session, patients and clients return home with verbal or pictorial instructions to follow. Unfortunately, these instructions are static and can be difficult to comprehend and comply with. To address the above problem, we propose an interactive avatar-based Virtual Reality platform, which enables individualized user training for a range of activities from home. Although there exist other avatar based training systems, our system provides real-time guidance rather than just providing scores, rendering our system unique. This feature allows the system to cater to the abilities of the user and to react to the user's performance by demonstrating the necessary adjustments to establish optimal conditions. In essence, our system is dynamic, allowing every user experience to be distinct.

Our avatar-based training system comprises of two sessions: (1) during an offline session, experts are recorded performing a specific set of activities with the trainee and their recorded data is used to render an avatar that serves as the user's instructor later, and (2) during a live home session, users select an activity and follow the instructions given by the pre-recorded avatar. Further, our system employs the MS Kinect [12] to capture the user's activity and gather the necessary data, which is delivered to and processed by the avatar training system.

Fig. 1. The proposed system

To determine the user's accuracy in performing his or her chosen application, we compare the user generated motion data with the pre-recorded avatar motion data. However, as shown in Figure 1, there may be a human reaction delay - the time it takes for the user to react to the avatar frame shown on the screen. Because of the existence of human reaction delay, it is inaccurate to compare the user motion data received by the platform with the motion vector corresponding to the current avatar frame, F, being rendered. Rather, the user motion data should be compared with an earlier frame, F-i, where i is the time shift of i frames needed for proper data alignment. Figure 2 illustrates the motion data misalignment predicament. The problem is not trivial and cannot be easily addressed by techniques such as time-stamping the data and comparing it accordingly for the following reasons.

1) Physical therapy consists of a group of activities. By time-stamping, we need to know where the activity begins, but the Kinect sensor only captures motion and is not aware of such information.

2) Human reaction delay changes from time to time, so even if the timestamps start at the same time, they will drift as time accumulates without correction.

Thus, to address possible data misalignment, we compare and analyze four time delay estimation methods, an existing method and three proposed methods, as pre-processing functions that align the user motion data with the proper avatar frame. The above allows accurate monitoring of user performance with regard to avatar instructions. To quantitatively evaluate the user's performance, and provide real-time guidance to the user in an effort to enhance user adherence to the avatar training, we develop a tolerance threshold level such that an accuracy level below the threshold will demand real-time feedback and an accuracy above the threshold will be tolerated.

In summary, the rest of this paper is organized as follows: In section 2, we examine related work. In section 3, we elaborate on the four methods to estimate user latency. The results of each method and a comparison between the approaches are also presented in this section. In section 4, we explain in detail our method in determining an activity's tolerance threshold. Lastly, in section 5 we conclude the paper and propose future work.

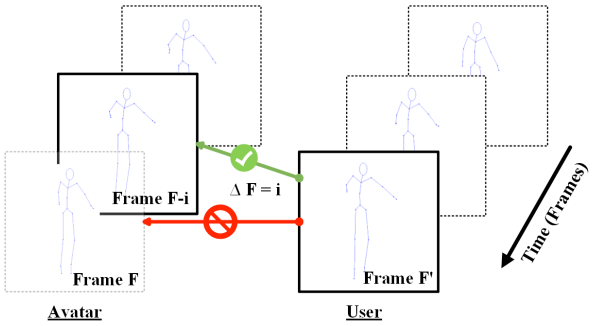

Fig. 2. User-avatar motion sequence misalignment caused by user delay; note that ΔF = i represents the delay between the sequences

II. Related Work

Previous work has been done in determining the appropriate time-shift, or delay, for proper data alignment. In particular, an existing time delay estimation method proposed by [1], calculates the time-shift between two sequences by determining the modulus of the quaternionic cross-covariance for all joints. However, as we will demonstrate in this paper, this approach is ineffective as it assumes the time-shift to be invariant. Thus, we present three new data alignment methods to account for a varying delay. In addition, [1] provides visual feedback to the user by displaying the user's calculated correlation-based score. As previously noted, we expand upon existing avatar based training systems by providing real-time guidance to the user as opposed to only providing scores, which merely indicate the accuracy of the user's performance but do not offer the necessary adjustments to correct existing inaccuracies. Furthermore, it is impossible for the user to impeccably duplicate the avatar's instructions. Thus, rather than potentially inundating the user with continuous visual feedback, we introduce a threshold tolerance level that alerts the system to only provide real-time guidance if the user is performing at an unacceptable level. In terms of quantitatively evaluating the user's performance, we adopt the metric proposed by [2], which calculates the user's accuracy by comparing the angles of the joints of interest between the subjects. Thus, our work builds upon the methods proposed by [1] and [2] by using the joint angles as a dataset for comparison with an appropriate tolerance threshold value for each application to provide beneficial real-time guidance to the user.

III. Time Delay Estimation

As previously mentioned, user delay, which is defined as the time difference or offset between the moment the user views the instruction and the moment when he or she correspondingly responds physically to the instruction, is not invariant. Thus, to accurately evaluate the user's performance, the temporal misalignment, which is the time-shift corresponding to the motion sequences of interest between the user and the avatar, must be corrected. As a result, in an effort to align the user's motion sequence with that of the instructor avatar's, we compare four distinct signal processing methodologies to estimate the time delay. However, prior to any time-shifting, we first pre-process the data by aligning both the user and avatar root joint positions, located at the center of the hip, to the 3-dimensional coordinate space origin, allowing for both a better comparison of the motion sequences and a reduction of noise caused by the Kinect. Furthermore, Smartbody [11], the open source character animation platform we developed upon, accounts for body size differences between the user and the avatar by retargeting the joint positions. In order to determine the estimated time-shift, we calculate the cross correlation given by

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f^*[m]g[m+n] \quad (1)$$

where f and g are the two discrete time real signals of interest, representing the avatar and user joint position motion data, respectively. The estimated time delay is then computed as

$$\tau_{delay} = \arg\max((f * g)[n]) \quad (2)$$

Because the Kinect captures 20 joints with an x, y, z component for every joint, we find the delay by determining the argmax of the sum of the cross correlations between each joint coordinate

$$(f * g)[n] = \sum_{i=1}^{K} \sum_{m=-\infty}^{\infty} f_i^*[m]g_i[m+n] \quad (3)$$

where K = 60 is the total number of joints and their corresponding coordinates, f and g are the K × N matrices of the joint values over the total number of frames N, and subscript i corresponds to the row index of matrices f and g.

A. Gesture Recognition

To implement the third and fourth time delay estimation methods, we introduce a gesture recognition program. For the purposes of our system, we define a gesture as a sub-motion of a complete motion sequence, i.e., a gesture is dynamic and characterized by a motion trajectory over a subsequence of frames. For example, in an activity where the user is to raise his or her hands into the air, touch his or her toes, and then spread his or her arms to the side, the described activity can be defined by and divided into three different gestures. Because gesture identification falls under a classification problem, we use a multi-class support vector machine (SVM) with a Gaussian radial basis function using the libSVM library [10]. Our feature vectors contain information on data including, but not limited to, the joint positions, joint velocities, joint angles, joint segment forces, joint segment momentum, and joint segment kinetic energies; the last three features were adopted and calculated using the equations presented by [4]. Furthermore, to avoid overfitting, we implement a grid search using k-fold cross validation to obtain the optimal regularization parameter, C, and bandwidth parameter, γ, to establish the optimal kernel. Thus, to classify a gesture, we compare the feature vector of the user's motion subsequence against previously generated feature vectors, which correspond to the pre-recorded avatar. In Table 1, we present the recognition results for the user in the form of a confusion and probability matrix of a motion sequence consisting of three gestures: (1) shoulder flexion (2) shoulder abduction (3) shoulder extension. This motion sequence will be the

experimental motion sequence and activity for all of the experiments in section 3. In Table 1, the rows correspond to the actual class and the columns correspond to the predicted class. Note that classes 1-3 correspond to the gestures 1-3, respectively. Further, the confusion matrix represents the number of true positive, true negative, false positive, and false negative errors and the probability matrix represents the probabilities of the predicted classes. From Table 1, we conclude that the system correctly recognized the user's gestures.

Table 1. Confusion and Probability Matrices

	Pred. 1	Pred. 2	Pred. 3
True 1	1	0	0
True 2	0	1	0
True 3	0	0	1
	Pred. 1	Pred. 2	Pred. 3
True 1	.9968	.0048	.0071
True 2	.0045	.9703	.0099
True 3	.0000	.0039	.9969

B. Method One: Existing Method

For the first method, we estimate a single overall delay of the particular activity by comparing the entire motion sequences of the user and avatar, solely using Equation 3 without any signal processing on the matrices f and g, the avatar and user motion sequences, respectively. This technique is similar to the method in [1]. A single, global time-shift is subsequently applied to the entire data set for all of the joints and for the entire activity period. Figure 3 displays the estimated time-shift between the user and avatar motion sequences for the activity described in section 3.1 using a preliminary implementation of our platform. Note that the peak of the plot corresponds to the time-shift value with the highest probability, representing the estimated user delay. As we will see in Figure 4(b), this method proves to be inaccurate. As mentioned previously, user reaction delay is not invariant. Thus, it is insufficient to estimate a single overall delay and apply a single time-shift to the motion data.

C. Method Two: Spatial Segmentation

In many exercises, the user is required to use multiple body parts at the same time. In these circumstances, it is quite possible that the user may end up having different delays for the different body parts. Thus, in the second method, we spatially segment the user and avatar bodies into five different parts: the head and torso, right arm, left arm, right leg, and left leg. The motivation behind our choices for the specific segmented body parts stems from the parent-child relationship between the joints [4]. Essentially, each child joint inherits the motion of its corresponding parent joint [5]. For instance, in the case of a shoulder rehabilitation exercise where the user lifts his or her right arm, the right wrist will inherit the momentum of the right elbow, which inherits the momentum of the right shoulder; simultaneously, each joint may possess its own momentum as well. As a result, we must include every joint of the arm to consider the arm as an independent body segment. A similar argument may be applied for the other body segments. Using the joint positions of each respective body part, we then estimate the delay of the entire temporal sequence for each body segment. That is, rather than using all

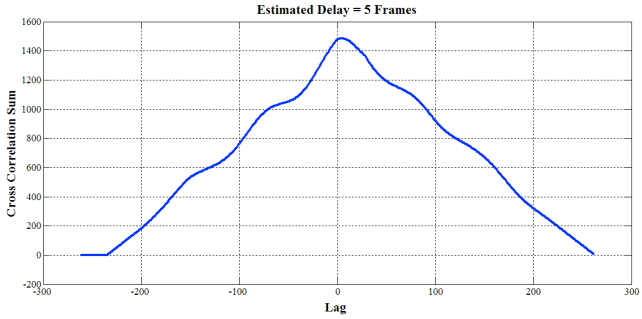

Fig. 3. Estimated time-shift employing existing method 1

60 joint values of the entire temporal sequence for time-shift estimation, we divide the 60 independent joint values into five distinct groups or body segments. Then, still using the entire temporal sequence, we find the time-shift of each body segment for a total of five individual delays and shift each body segment by its appropriate value. Our results in Table 2, which use the activity described in sections 3.1 and 3.2, demonstrate that our hypothesis was correct in that different body parts can possess unique delays. Note that each numeric value in Table 2 represents the estimated time-shift value for a particular body segment. The experimental results we present in section 3.6 will show that the second method performs better data alignment than the existing method. However, even though the second method incorporates spatial considerations, it still fails to account for a temporal varying delay.

Table 2. Estimated time delays (in frames) employing method 2

	Torso	R. Arm	L. Arm	R. Leg	L. Leg
Delay	-2	-5	-5	0	-11

D. Method Three: Temporal Segmentation

Similar to the argument made in the preceding paragraph that different body parts can possess different delays, the motivation behind the third method stems from the fact that the delay varies with respect to time as well. Thus, we propose a new method to account for this predicament. In essence, the third method executes temporal segmentation by means of gesture recognition. As previously mentioned, this method involves machine learning. When a gesture is identified, that particular temporal window is extracted from the entire temporal sequence. The delay of each temporal window is then calculated, resulting in an individual delay estimate for each individual gesture. In other words, we still use all 60 joint values for processing, but rather than using the entire temporal sequence, we break the temporal sequence into subsequences. Our results in Table 3 demonstrate that different gestures can possess unique delays. Note that each numeric value in Table 3 represents the estimated time-shift value for a particular gesture. As a result, method three proves to possess a desirable trait for accurate latency estimation. However, similar to the first method, method three fails to account for a spatial varying delay.

Table 3. Estimated time delays (in frames) employing method 3

	Flexion	Abduction	Extension
Delay	8	-9	-4

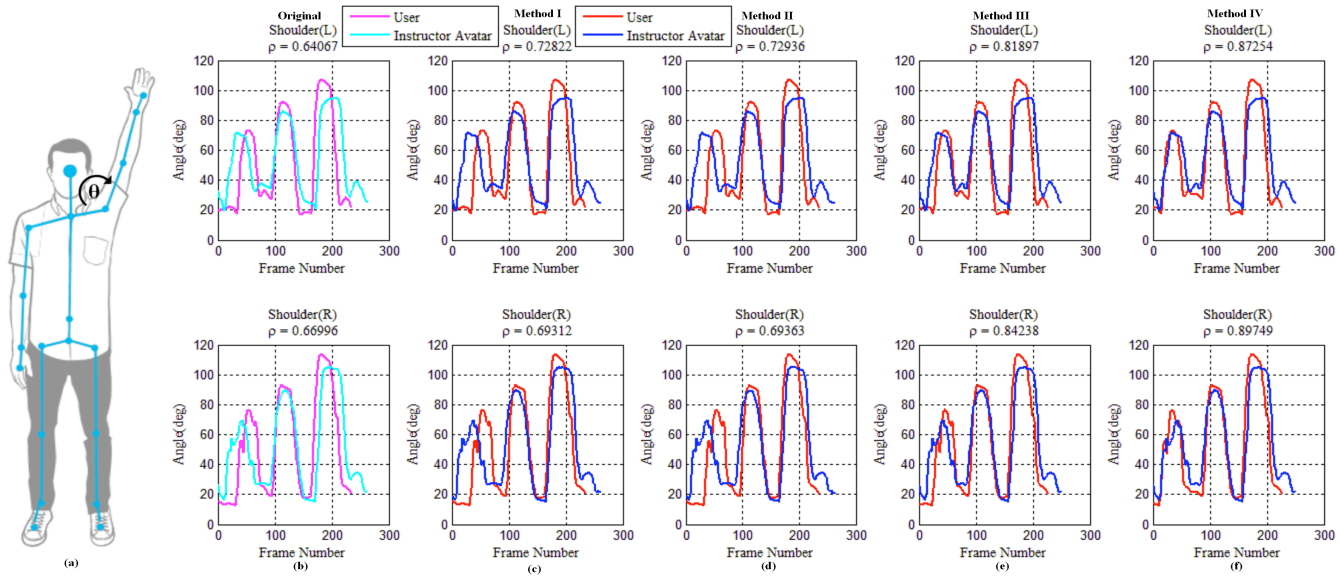

Fig. 4. (b)-(f) displays the outputs after employing the four time estimation methods using the Spearman rank correlation coefficient as a metric to identify which technique best aligns the motion datasets between the user and avatar. Because the experimental activities target the upper body, we use the left and right shoulder angles show in (a) as the motion dataset used for analysis.

E. Method Four: Spatio-Temporal Segmentation

Our fourth method builds upon the second and third methods in accounting for both a spatial and time varying delay. Essentially, the fourth method conducts both spatial and temporal segmentation by dividing the body into five unique segments and recognizing gestures, respectively; we not only group all of the joint values into distinct segments, but we also divide the temporal sequence into separate temporal windows. Thus, the overall process works as the following: The entire temporal sequence is segmented by gestures, as recognized by the system, and converted into a series of temporal windows. Each subsequent temporal window is divided in the spatial space into five unique body parts. The delay of each body part of each temporal window is then calculated and the data set for each segment per gesture is shifted by its appropriate amount. Table 4 displays the results of applying method four for the same sample training session used throughout section 3. Note that each numeric value in Table 4 represents the estimated time-shift value for a particular body segment per gesture.

Table 4. Estimated time delays (in frames) employing method 4

	Torso	R. Arm	L. Arm	R. Leg	L. Leg
Flexion	10	11	8	0	0
Abduction	-8	-9	-15	-11	-15
Extension	-4	-5	0	0	0

F. Results

In this subsection, we compare the effectiveness of each time estimation method. From our supplementary video, one can see the effects of user delay at 12 seconds by the misalignment between the user avatar, reflecting the activity of the user captured by the MS Kinect, and the pre-recorded instructor avatar. Figure 4 (b)-(f) displays the results where the blue/cyan plots represent the avatar and the magenta/red plots represent the user. Further, the x-axis represents the time in frames and the y-axis represents the left and right shoulder joint angles. The joint angles can be calculated by

$$\cos(\theta) = \frac{x^T y}{\sqrt{x^T x} \cdot \sqrt{y^T y}} \tag{4}$$

where the vectors x and y represent the reference vectors, or body segments. To clarify, the reference vectors of an elbow angle would be defined as the shoulder-elbow and elbow-wrist vectors. Further, each column – with the exception to the first column, which represents the original joint angle sequences of the respective subjects – is the output after applying each method. We adopt the Spearman Rank Correlation Coefficient as a metric to determine which method produces the best motion sequence alignment. The Spearman rank correlation coefficient is given by

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2 - 1)} \tag{5}$$

where n is the sample size and $d_i = x_i - y_i$ is the difference between the ranks x_i and y_i determined from the raw scores X_i and Y_i; for our purposes, n is the number of frames of the sequence and the raw scores X_i and Y_i correspond to the user and avatar joint angles, respectively. Our results demonstrate that the fourth method, which conducts spatio-temporal segmentation, is the most effective, as it possesses the highest correlation coefficient value. By visual inspection, we can also identify that the fourth method, as seen in Figure 4(f), best aligns the joint angles over time.

To further cement the fourth method as the optimal latency estimation approach, we conduct multiple trials on different activities using different joint angles and display the results in Figure 5. The results are presented as histograms whereby the y-axis represents the number of trials and the x-axis represents

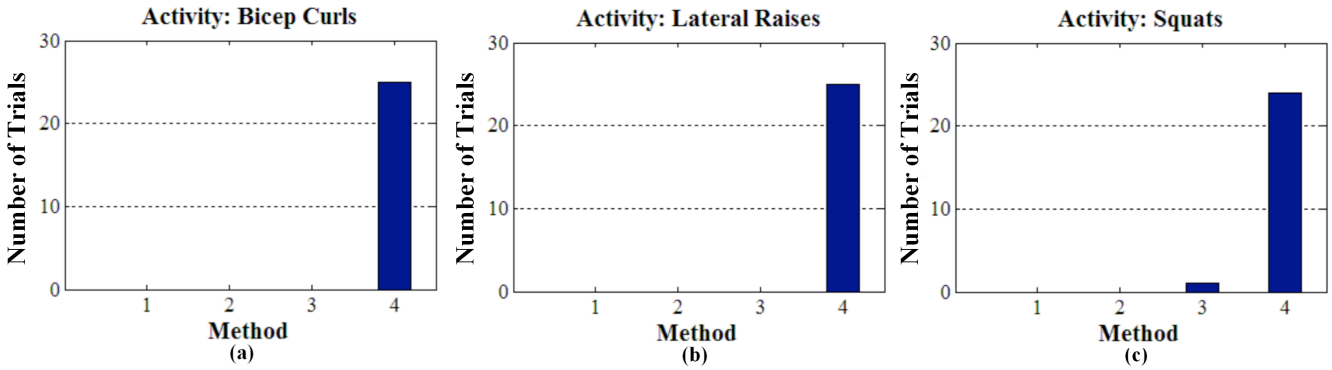

Fig. 5. The figures above plot the results of comparing the four time estimation methods for three activities: (a) bicep curls, (b) lateral raises, and (c) squats

the four distinct time estimation approaches. For this particular experiment, we conducted 25 trials per activity and plotted the number of times a particular method produced the highest Spearman correlation coefficient value. The three activities used in the experiment were as follows: bicep curls, lateral raises, and squats. The motion datasets corresponding to the three activities were elbow angles, shoulder angles, and knee angles, respectively. From all of the experimental results under this section, we conclude that the fourth method performs the best data alignment as it best accounts for a spatial and time varying delay.

IV. Determing User Accuracy

To assist the user in accurately performing their chosen activity, we provide real-time visual feedback and guidance to the user's device. According to [3], real-time feedback can also potentially assist the user in avoiding injuries. As previously noted, we compare the user's motion data with that of the avatar's, which we treat as the ground truth. Because it is impossible for the user to achieve perfect accuracy, we develop a threshold tolerance level, such that an accuracy level below the threshold will demand real-time feedback and an accuracy above the threshold will be tolerated. To determine the desired threshold value, we conduct an experiment with the assistance of a professional personal trainer.

A. Experiment Setup

After collecting the dataset of timestamps, we process the motion data offline to determine the threshold level of accuracy for different activities. To find the threshold, we first execute our spatio-temporal algorithm, method four, to properly align the motion datasets and to also properly shift the timestamps accordingly. Next, we create a vector dataset containing the differences in joint angle values between the user and avatar at the shifted timestamps. Subsequently, we create a histogram of the vector set. Because histograms vary with respect to the bin size, we employ the Shimazaki-Shinomoto histogram binwidth optimization method to find the optimal number of bins for better distribution estimation [7]. Finally, we fit the best distribution to the dataset and estimate the distribution parameters by using the Bayesian information criterion (BIC)

$$BIC = -2 \cdot \ln p(x \mid \theta, M) + k \cdot (\ln(n) - \ln(2\pi)) \qquad (6)$$

where x is the vector dataset, n is the number of frames, k is the number of estimated free parameters, and $p(x|\theta, M)$ is the maximized value of the likelihood function of the model M with θ being the set of parameter values that maximizes the likelihood function [8].

At the same time, we create an "inverse" histogram, which represents the frequency at which the personal trainer did not mark the times when the same joint angle difference values were detected. Again, we fit the best possible continuous probability distribution to the "inverse" histogram using the method described above. Figures 7(a)-(b) under section 4.3 depicts both histograms with their respective continuous probability distributions for the shoulder press exercise.

Using both distributions, we now attempt to find the optimal threshold x_{opt}. In doing so, we establish our decision function $\alpha(x)$, which enables the system to accurately classify the user performance as one of two categories ω: intolerable ω_1 or tolerable ω_2. To reach this threshold value, we minimize the overall risk

$$R = \int R(\alpha(x) \mid x) P(x) dx \qquad (7)$$

by selecting the action α_i for i = 1,2 that minimizes its associated conditional risk

$$R(\alpha_i \mid x) = \sum_{j=1}^{M} \lambda(\alpha_i \mid \omega_j) P(\omega_j \mid x) \qquad (8)$$

where M = 2 is the number of classes and $\lambda(\alpha_i|\omega_j)$ is the element in the loss matrix representing the cost of selecting class ω_i when the true class is ω_j. Further, we assume our loss matrix to be biased towards type II errors (false negatives), i.e., $\lambda(\alpha_2|\omega_1) > \lambda(\alpha_1|\omega_2)$. Thus, our loss matrix is defined as

$$L = \overbrace{\begin{bmatrix} 0 & \lambda(\alpha_1, \omega_2) \\ \lambda(\alpha_2, \omega_1) & 0 \end{bmatrix}}^{\textit{Intolerable} \quad \textit{Tolerable}} \qquad (9)$$

whereby the columns represent the two classes and the rows correspond to the two system actions α: providing real time guidance α_1 and tolerating the user's performance α_2. In other words, we assume the loss incurred for classifying the user's performance as tolerable when the user is inaccurately following the avatar to be greater than the loss incurred for

classifying the user's performance as intolerable when the user is accurately following the avatar. To compute the conditional risk given by Equation 10, we apply Bayes formula

$$P(\omega_j \mid x) = \frac{P(x \mid \omega_j)P(\omega_j)}{P(x)} \tag{10}$$

to find the posteriors $P(\omega_j|x)$ given the known prior probabilities $P(\omega_j)$ and conditional probabilities, or likelihood, $P(x|\omega_j)$ for the two categories. The evidence can also be calculated as

$$P(x) = \sum_{i=1}^{M} P(x \mid \omega_i)P(\omega_i) \tag{11}$$

For our purposes, the prior probabilities $P(\omega_j)$ are assumed to be equiprobable: $P(\omega_k) = 0.5$, k=1,2. We adopt this equiprobable assumption to account for varying priors. Specifically, consider the following two cases: (1) users who use our proposed system are assumed to perform to the best of their abilities with the intent to correctly learn the chosen application, thus we assume $P(\omega_2) > P(\omega_1)$ (2) users who select new activities may be more prone to errors, thus we assume $P(\omega_1) > P(\omega_2)$. Thus,, we assume equiprobable priors

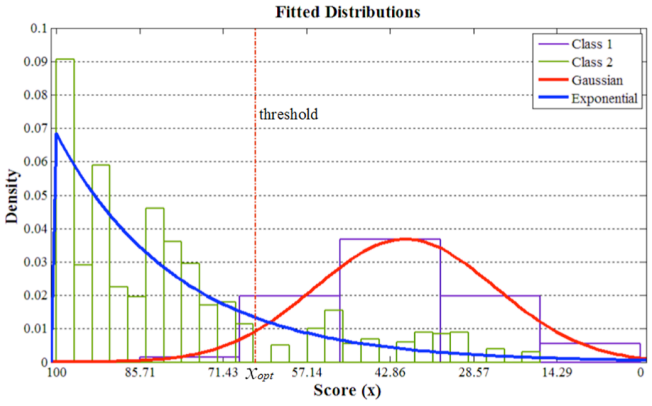

Fig. 6. This figure displays the two histograms, representing the intolerable class ω_1 and tolerable class ω_2, with their associated distributions $P(x|\omega_1)$ and $P(x|\omega_2)$ respectively. The dashed line represents the shoulder angle threshold x_{opt} for the shoulder press exercise.

to balance the described scenarios.

Further, each joint angle difference (the histogram bin variable) between the user and the avatar will receive a correlation-based score that is a real, scalar value $x \in \Re$ ranging from 0-100. Note that small differences between the joint angles of the user and avatar correspond to higher scores. Thus, our conditional probability densities, the best fit distributions to the two histograms, can be described as $P(x$=score$|\omega_1$=*intolerable*) and $P(x$=*score*$|\omega_2$=*tolerable*). Finally, we obtain the optimal threshold x_{opt} for a particular activity by solving the equation

$$P(x_{opt} \mid \omega_1)\lambda_{21}P(\omega_1) = P(x_{opt} \mid \omega_2)\lambda_{12}P(\omega_2) \tag{12}$$

where $\lambda_{ij} = \lambda(\alpha_i|\omega_j)$. As a result, we can employ the likelihood ratio

$$\frac{P(x \mid \omega_1)}{P(x \mid \omega_2)} > \frac{\lambda_{12}}{\lambda_{21}} \frac{P(\omega_2)}{P(\omega_1)} \tag{13}$$

to classify the performance as ω_1 if the inequality is satisfied, i.e., the likelihood ratio exceeds the threshold x_{opt}, and classify the performance as ω_2 otherwise [9]. Note that the score x decreases in magnitude from left to right, as seen in Figure 6, while the difference in joint angles increases from left to right.

B. *Results*

In this section, we present our results in determining the threshold values for three activities: the shoulder press, altering shoulder rotation, and oblique engagement. To illustrate the method in determining the threshold described in section 4.2, we provide the following example for finding the shoulder joint angle threshold for shoulder presses. Employing the Shimazaki-Shinomoto histogram binwidth optimization method, we find the optimal bin size for class one's histogram to be 5 bins and the optimal bin size for class two's histogram to be 27 bins. Furthermore, class one's histogram is fitted to a normal distribution characterized by N(41.821°,117.538°) while class two's histogram is fitted to an exponential distribution with the rate parameter $\beta = 0.0684$ using the BIC. Thus, the shoulder angle threshold x_{opt} is found by using Equation 14 with the appropriate likelihoods and solving

$$\lambda_{21} \cdot \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x_{opt}-\mu)^2}{2\sigma^2}} = \lambda_{12} \cdot \beta e^{-\beta x_{opt}} \tag{14}$$

given $\lambda_{21} = 1.5$ and $\lambda_{12} = 1$. Solving this equation yields the threshold $x_{opt} \approx 67$, which is equivalent to an angle difference of 23.655°. In other words, a shoulder angle difference less than 23.655° is tolerable and a shoulder angle difference greater than 23.655° demands real-time guidance. The threshold values for activity two and three are $x_{opt} \approx 84$ (elbow angle) and $x_{opt} \approx 78$ (hip angle), respectively.

V. Conclusion and Future Work

From our results, we identify the fourth method, which executes spatio-temporal segmentation, as the optimal approach to estimating user delay. Employing this method allows us to properly shift the datasets to accurately compare the user and avatar motion sequences at the correct frames. Future work will be dedicated to developing our data alignment method to also consider the possibility of the CP following the avatar faster or slower by identifying potential differences in speed for certain time periods and adjusting the CP sequences by the ratio of the speed difference prior to any time-shifting. Furthermore, the results of our experiments demonstrate that a unique and effective threshold value can be obtained for each exercise. However, because the recorded timestamp values obtained from the personal trainer may vary with other personal trainers, our timestamp dataset can be characterized as a subjective dataset. Thus, a larger dataset from a greater range of professionals/experts may be necessary to determine a more widely accepted threshold value. Furthermore, we calculated our threshold values under the assumption that a type II error was more detrimental than a

type I error, i.e. $\lambda(\alpha_2|\omega_1) > \lambda(\alpha_1|\omega_2)$, and under the assumption that the prior probabilities were equiprobable $P(\omega_2) = P(\omega_1)$. Further examination and experimentation of these assumptions may help in optimizing our threshold values. Lastly, our current system employs the original Microsoft Kinect, which suffers from a myriad of issues, especially in accurately acquiring data from the subjects; this data corruption is most notable in particular exercises where skeletal merging or joint overlap occurs. As a result, we intend to substitute the original Kinect for the new Microsoft Kinect2 in an effort to process more accurate data.

REFERENCES

[1] Alexiadas, Dimitrios, Petros Daras, Tamy Boubekeur, Philip Kelly, Noel O'Connor, and Maher Ben Moussa. "Evaluating a Dancer's Performance using Kinect-Based Skeleton Tracking." *MM '11 Proceedings of the 19th ACM international conference on Multimedia*: n. pag. Print. Dennis R. Morgan, "Dos and don'ts of technical writing," *IEEE Potentials*, vol. 24, no. 3, pp. 22-25, Aug. 2005.

[2] Da Gama, Alana, Thiago Chaves, Lucas Figueiredo, and Veronica Teichrieb. "Improving Motor Rehabilitation Process through a Natural Interaction Based System Using Kinect Sensor." *IEEE Symposium on 3D User Interfaces 2012*: n. pag. Print.

[3] Chang, Chien-Yen, Belinda Lange, Mi Zhang, Sebastian Koenig, Phil Requejo, Noom Somboon, Alexander Sawchuk, and Albert Rizzo. "Towards Pervasive Physical Rehabilitation Using Microsoft Kinect."*International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth) and Workshops*: n. pag. Print.

[4] "Joint Orientation." *Joint Orientation*. Web. <http://msdn.microsoft.com/en-us/library/hh973073.aspx>.

[5] Kahol, Kanav, Priyamvada Tripathi, Sethuraman Panchanathan, and Thanassis Rikasis. "Gesture Segmentation In Complex Motion Sequences." (2003): 105-08. IEEE. Web. 1 July 2014.

[6] Livingston, Mark A., Jay Sebastian, Zhuming Ai, and Jonathan W. Decker. "Performance Measurements for the Microsoft Kinect Skeleton." *Virtual Reality Short Papers and Posters (VRW)* (2012): 119-20. Print.

[7] Shimazaki, Hideaki, and Shigeru Shinomoto. "A Method for Selecting the Bin Size of a Time Histogram." *Neural Computation* (2007): 1503-527. Print.

[8] Cavanaugh, Joseph E. "171:290 Model Selection Lecture V: The Bayesian Information Criterion." University of Iowa, 25 Sept. 2012. Web.

[9] Duda, Richard O., and Peter E. Hart. "Bayesian Decision Theory." *Pattern Classification*. 2nd ed. New York: Wiley, 2001. Print.

[10] Chih-Chung Chang and Chih-Jen Lin, LIBSVM: a library for support vector machines. ACM Transactiosn on Intelligent systems and Technology, 2:27:1--27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[11] http://smartbody.ict.usc.edu

[12] www.xbox.com/en-US/kinect