

A Recurrent Neural Network Approach to Virtual Environment Latency Reduction

Aaron Garrett
Knowledge Systems Laboratory
Jacksonville State University
Jacksonville, AL 36265
aarong@ksl.jsu.edu

Mario Aguilar, Ph. D.
Knowledge Systems Laboratory
Jacksonville State University
Jacksonville, AL 36265
mariao@ksl.jsu.edu

Yair Barniv, Ph. D.
NASA/Ames Research Center
Human Factors Research Division
Moffet Field, CA 94035-1000
ybarniv@mail.arc.nasa.gov

Abstract – We present a recurrent neural network system designed to predict future angular acceleration of the human head from current angular acceleration data. These predictions can be used to supplement head tracking in virtual environments in order to reduce latency and increase tracking accuracy, thus enhancing the user’s performance and comfort.

I. INTRODUCTION

With increasingly smaller and faster computing machinery, the use of virtual environments (VE) in everyday applications is becoming more a matter of fact, rather than a matter of science fiction. However, using virtual environments is not without its difficulties. One of the most notable problems facing current VE applications is the perceptible latency that is experienced by the user as the computer displays are updated. Such perceptible latency has been shown to have undesirable effects on users of virtual environments, including a lack of accuracy during tracking tasks, motion sickness, and loss of immersion [1]. Ellis, et al, have shown that visual latency, more than spatial sensor distortion or low update rates, is the primary cause of RMS tracking errors [2]. This latency comes in three different forms – rendering lag, application-dependent processing lag, and user input device lag [3]. Rendering lag is due to the amount of time it takes for the computer screen to be updated. Application-dependent processing lag is the latency that arises from the computation of the three-dimensional model. This latency is entirely dependent on the complexity of the model. Finally, user input device lag is the lag introduced from the necessary communication between the position and orientation tracking system and the VE application. Because of its relevance to our current work, in this paper we will only consider this final type of lag.

User input device lag can range between 10ms and 120 ms [3]. Because of the detrimental effects that often result from lag in virtual environments, many motion trackers implement a prediction mechanism within the tracker itself that attempts to compensate for the device lag. This prediction mechanism is often no more than a crude averaging and extrapolation from the current movement to the next movement in time. While such a scheme can be useful and effective in certain situations, it manifests serious problems if the user moves quickly or changes velocity. Thus, in order to avoid experiencing the cyber-sickness caused by lag, the user must make slow, deliberate movements [4]. Also, using these approaches, predictions out to a useful range (i.e., 20–40 ms) are subject to very large errors. If differential acceleration data (as generated by linear

accelerometers or rate gyros) is used, more accurate extrapolation systems can be created because of the higher resolution of the original data. We will discuss these extrapolation methods in more detail in Section III.

In an effort to improve upon the prediction within the tracker, itself, other prediction mechanisms have been implemented which make use of Kalman filters. For instance, Friedmann, et al, used Kalman filters to predict the position, velocity, and rotations of a Polhemus sensor [4]. Their results show that accurate predictions can be made except at points where the user changes acceleration and/or direction very sharply. Such unexpected changes were also the major cause of problems for the on-board prediction method of the trackers themselves. Friedmann, et al, dealt with this problem by using a multiple model approach. In this approach, several different Kalman filters were created to accommodate different types of user movements. At each instance of prediction, the filter with the maximum likelihood was used.

While the above study dealt with both positional prediction and rotational prediction, it has been shown that users are much more affected by rotational latency than positional latency. This has been attributed to two main causes. First, because of the limitations of most motion trackers, positional changes are limited to a small range. Thus, users tend to make such changes infrequently. Second, rotational changes generally cause more noticeable changes in the scene than do positional changes [5]. Because of this fact, it seems reasonable to focus current motion prediction research to deal with only the rotational accelerations of the user. In fact, Ellis, et al, carried out an experiment in which Kalman filters were used to predict only the angular accelerations of user head motions within a virtual environment [6]. These angular accelerations are based on approximations using the Polhemus tracker measurements, not on actual rate gyro data which would be much more accurate.

Using differential acceleration data collected from rate gyros is indeed an extremely powerful approach to head movement prediction. However, to the best of our knowledge, linear approximation techniques such as Kalman filters have only been applied to approximated acceleration data. In an effort to make use of collected differential acceleration data and to move beyond the linear filtering techniques that have been applied to date, we present a recurrent neural network approach to the problem of angular acceleration prediction. However, rather than using angular acceleration about all three axes, we constrain the problem by using angular acceleration about the z-axis only. (This would

be rotation about the vertical.) This constraint was made to allow us to focus more closely on the other aspects of the work, such as choice of network parameters and types of error measurements. Also, because users within virtual environments perform such rotations more often than rotations about the x- or y- axes, most of the information about user motion is carried by this variable. Because of this, a neural network trained on z-axis rotational data could be easily extended to encompass the x- and y- axis information as well. In this way, where many different Kalman filters may be needed to accurately model a majority of user head motions, one neural network would suffice.

The layout of the remainder of the paper is as follows. In Section II, we discuss the experimental setup for data collection and outline the particular neural network architecture chosen. In Section III, we present our findings along with a discussion of the error measurement that was used. We also contrast our results against a widely used prediction system embedded within a motion tracker. Finally, in Section IV we draw our conclusions.

II. METHODS

A. Experimental Setup

An InterSense IS-600-series motion tracker was used to collect angular acceleration data from two different subjects. The sampling rate for both subjects was set to 160 Hertz. Each subject was fitted with a helmet containing the inertial sensors and was instructed to perform movements from a pre-defined set. Such movements included abrupt movements from right to left; smooth, continuous movements from right to left; smooth movements from right to left with a pause in the center; and abrupt movements from right to left with a pause in the center. These movement types were chosen to be representative of head motion that would be likely within a virtual environment. For instance, the abrupt motions were chosen to be similar to startled movements, and the continuous motions were chosen to be similar to casual scanning movements.

Two other motions were also included that contained some rotation about the vertical along with rotation about the other two axes. These were movements in a clockwise circle (as if the nose were following the hand of a clock), movement along an upper-right to lower-left diagonal, and movements along an upper-left to lower-right diagonal. These movements were included in order to test the neural network’s ability to generalize to different movements.

After each experiment, the angular acceleration data collected was saved into a file for use with the learning system. In this way, all training could be done off-line without the need for more data collection.

B. Feature Description

No learning system is effective without information-rich inputs into that system. This means that the extraction of appropriate features that characterize the inputs is essential

Table 1
Summary of Features

Feature	Description
Moving Average	Average angular velocity within a 25 ms window
Average Slope	Average slope of angular velocity within a 25 ms window
Number of Zero-crossings	Number of instances where the angular velocity was 0 within a 25 ms window
Average Frequency	Instantaneous frequency at each time-step
Average Period	Average amount of time between zero-crossings within a 25 ms window
Cumulative, Non-overlapping Gradient	Cumulative measure of the gradient using non-overlapping 1.5 ms windows
Curve Complexity	Sum across instantaneous gradients within a 25 ms window

for success. The set of angular acceleration-based features that we used to produce the results discussed below are summarized in Table 1. The moving average, average slope, average frequency, and average period are self-explanatory. The number of zero-crossings at time t , Z_t , is a measure of the frequency of the curve. It is defined to be the number of instances where the angular velocity is 0 within a 25 ms window. It is defined by the following equation:

$$Z_t = \sum_{i=t-w}^t z(i), \text{ where } z(t) = \begin{cases} 1 & \text{if } a(t) = 0 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The curve complexity is a feature that attempts to measure the “shape” of the curve [7]. Equation 2 provides the definition of this feature (referred to as the “waveform” in [7]):

$$W(t) = \sum_{i=t-w}^t \left| |a(i)| - |a(i-1)| \right| \quad (2)$$

The cumulative, non-overlapping gradient was found to be the most useful feature for predicting the angular acceleration data. Equation 3 defines this feature:

$$G_{cum}(t) = \sum_{i=0}^{\lfloor \frac{t}{2w} \rfloor} a(t-2i) - \sum_{i=0}^{\lfloor \frac{t}{2w} \rfloor} a(t-i) \quad (3)$$

This feature provides a measure of total change at the current time from all past movements. It provides a running measure of change *per full movement*. This is because the measure “resets” itself whenever the angular acceleration reaches zero.

This is very useful when trying to predict on a movement-by-movement basis. (In each of these equations, the w term represents the window size.)

We found that the seven features presented above provided a good representation of the dynamics and characteristics of the raw angular acceleration data. As such, these features were normalized and used as the inputs into the recurrent neural network.

C. Network Architecture

A recurrent neural network known as an *Elman neural network* was used as our learning system. The Elman neural network is based on the back-propagation algorithm. It includes the typical feed-forward connectivity with the addition of a recurrent connection from the output of the hidden layer at time t to the input of the hidden layer at time $t+1$. This recurrent connection gives the network an exponential “memory” of past events [8]. This “memory” makes the Elman network very effective in learning time-sequential patterns, such as the type we see in angular acceleration data.

The Elman network was implemented in Matlab 6.1 using the Neural Network Toolbox. The appropriate parameters for the network, which include the number of hidden neurons, the training function, the transfer functions, etc., were determined experimentally. The final parameters chosen were as follows:

- Number of hidden neurons – 15
- Training function – BFGS quasi-Newton method
- Transfer functions – Hyperbolic tangent sigmoid function for hidden neurons; linear function for output neurons
- Search function – One-dimensional minimization using the method of Charalambous [9]
- Weights initialization function – Nguyen-Widrow layer initialization function

This set of network parameters was used to generate all of the data presented in the remainder of the paper.

III. RESULTS

A. Results of the Elman Network

We found that training the neural network with a single example of two movements was sufficient to achieve good generalization across the data set. We trained the network on an abrupt right-left movement from one subject along with a continuous right-left motion from a second subject. The network was then tested against a comprehensive set of representative movements from all subjects. The results of this test are shown in Figure 1. In this figure, the red solid curve corresponds to the actual rotational acceleration 20ms in the future. The blue dashed-dotted curve corresponds to the predicted angular acceleration as predicted by the neural network. Here, a successful prediction would show a large amount of overlap between the two curves. This

representation scheme was used for all of the figures presented within the paper, except where noted otherwise.

Figure 2 displays an enlarged image of the section of the curves contained within the solid black rectangle in Figure 1. In this figure, the qualitative degree of match between the two curves can be seen in greater detail. Clearly, the neural network is capable of predicting well even in those areas of abrupt changes in angular acceleration.

Figure 1. Results of Neural Network on Comprehensive Test Set. Acceleration (y-axis) during left-right movements is plotted against time (x-axis).

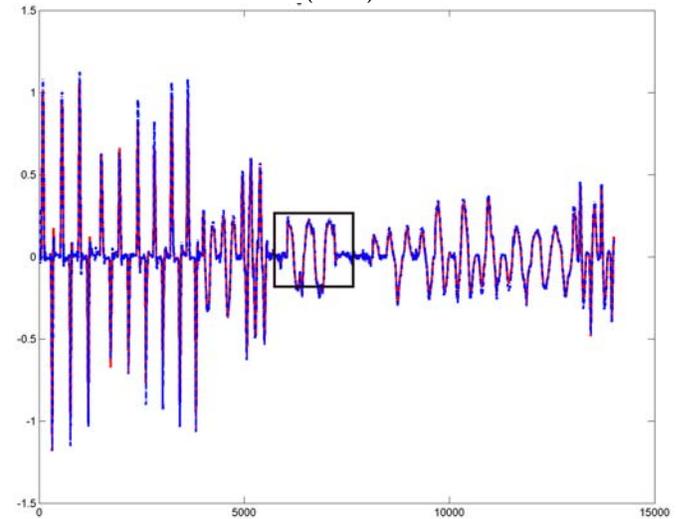
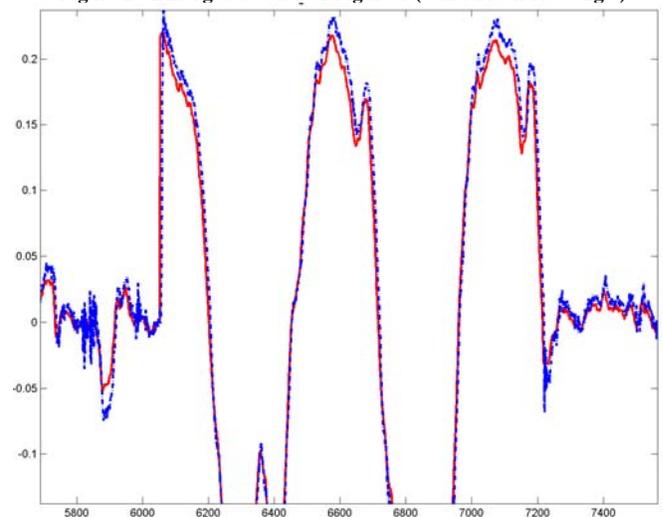


Figure 2. Enlarged Section of Figure 1 (Solid Black Rectangle)



In addition to qualitative evaluations of performance, we also used the Mean Squared Error (MSE) measurement to determine our predictive accuracy. For the data above, the MSE was 0.0357. This measure does match our qualitative assessment of the predictive ability of the neural network. However, the MSE is not entirely effective when assessing accuracy of prediction in the temporal domain as is required here. We will consider this point in more detail in the next section.

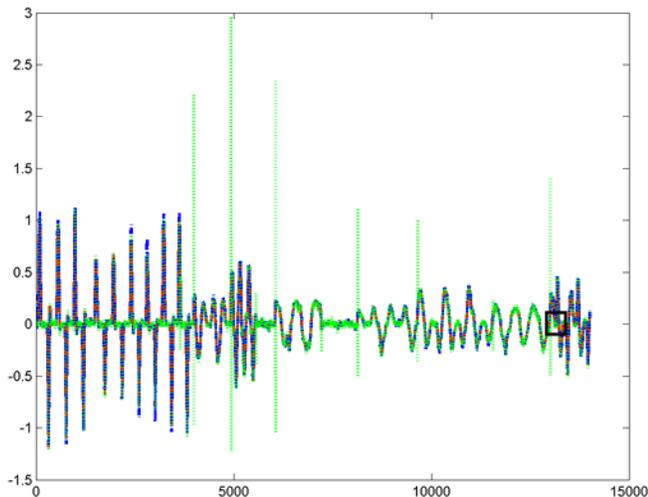
B. Comparison with Extrapolation Prediction

To the best of our knowledge, the procedure used within the InterSense IS-600-series trackers makes use of a second-order extrapolation scheme in order to predict future angular accelerations from current data [InterSense spokesman, personal communication]. This extrapolation is achieved using the following equation:

$$a(t_0 + n) = a(t_0) + na'(t_0) + \frac{1}{2}n(n-1)a''(t_0) \quad (4)$$

Here, t_0 is the current time point, and n represents the number of milliseconds beyond which we want to extrapolate. The equation presented above is based on second-order Taylor polynomial approximations. It should be noted that this extrapolation method amplifies noise, especially when the temporal extrapolation distance is high. To combat this problem, we introduced a slight smoothing by using a moving average with a small window size ($w = 5ms$). While this removed some of the noise in the resulting prediction curve, a large amount of noise remained.

Figure 3. Comparison of Extrapolation with Neural Network Prediction



The extrapolation function discussed above was applied to the comprehensive dataset mentioned in Part A. The angular acceleration data was predicted 20ms in advance, just as with the neural network approach. The qualitative results of this extrapolation are displayed in Figure 3. In this figure, the extrapolation curve was added to the existing curves from Figure 1. This curve can be seen as a dotted green curve in Figure 3 (as well as in all other figures, except where otherwise noted).

Figure 4 presents the detail of the highlighted section of Figure 3 (denoted by the black rectangle near the right side of the graph). It can be observed that the extrapolation curve contains large amounts of noise, whereas the neural network prediction has a much tighter match to the target curve. The noise generated by the extrapolation would be revealed

within a virtual environment as high-frequency jitter. Once again, as a more quantitative comparison, the MSE for the extrapolation was calculated to be 0.0502. This is significantly higher than the error generated by the neural network system, but as we mentioned earlier, the MSE is not sensitive to temporal pattern matches. In the context of this work, the temporal pattern of the curves carries a significant amount of the predicted information.

Figure 4. Enlarged Section of Figure 3 (Solid Black Rectangle)

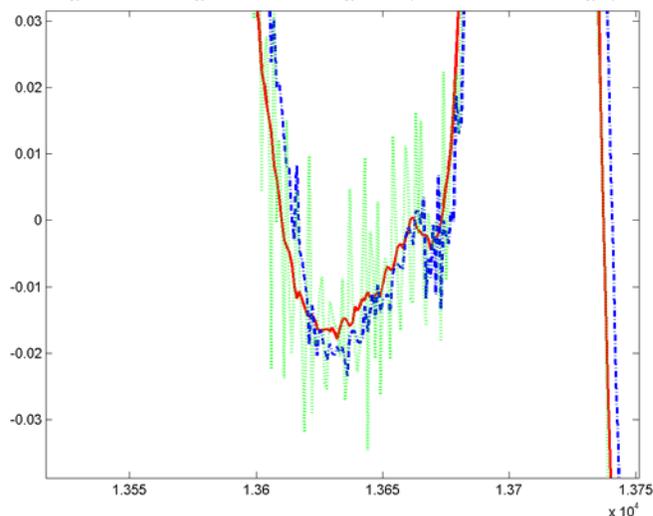
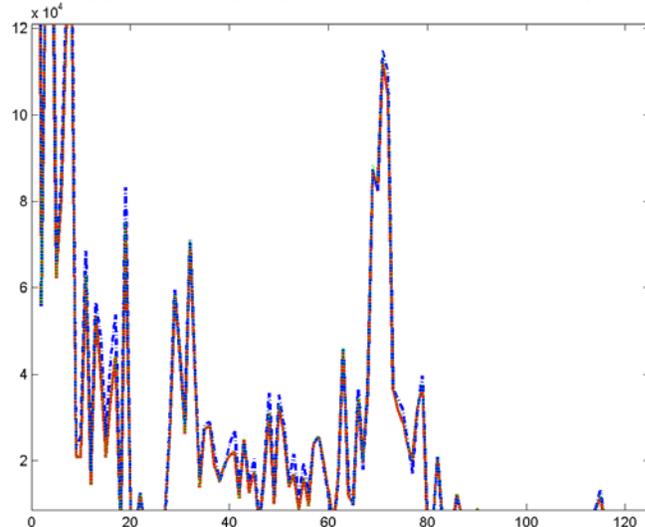


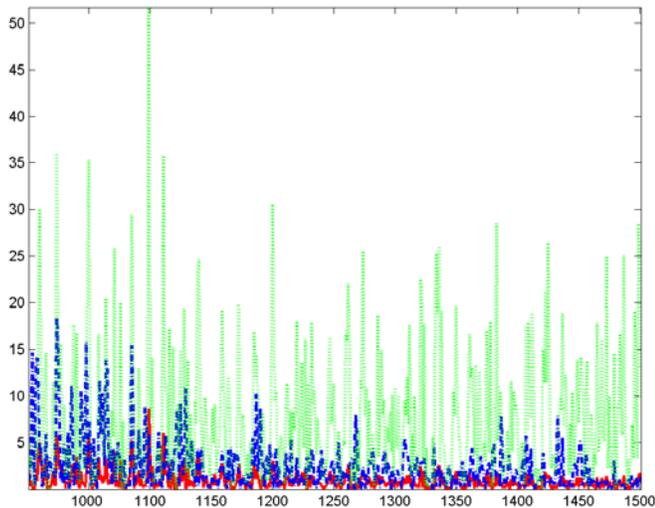
Figure 5. Comparison of Low Frequency Information. Energy (y-axis) vs frequency (x-axis, not in Hz due to subsampling)



In an effort to overcome a similar problem, Liang, et al, turned to the frequency domain to compare different curves [5]. We chose to do a similar qualitative comparison for our results. In this way we can qualitatively assess the results not only spatially, but also spectrally. Figure 5 displays the low frequency information of each of the three curves from Figure 3. Essentially, Figure 5 reveals that both the neural network prediction and the extrapolation match closely with the actual angular acceleration curve within the low

frequency domain. However, when we consider high frequency information, as in Figure 6, we see that the extrapolation procedure generates considerable error. It is within this range that abrupt acceleration changes would occur and where other methods have failed to produce accurate prediction. It can be observed that the neural network prediction achieves a significant improvement over the extrapolated results.

Figure 6. Comparison of High Frequency Information



IV. CONCLUSIONS

The Elman-based neural network system presented here has been shown to predict future angular velocities with a high degree of accuracy. The qualitative results show that, given a small representative sample of movements, the Elman-based system can effectively generalize to predict across subjects and movement types. When compared with the current extrapolation methods built into head-tracking devices, we have demonstrated that a neural network system tends to produce increased accuracy.

The need for quantitative measures of error in this type of domain is an important research issue. In the absence of a true measure of temporal error, we introduced another qualitative measurement based on the work of Liang et al [5] that makes use of the frequency analysis of acceleration curves. While this does make an even stronger qualitative statement about the high frequency error of an extrapolation system, it does not give us a true quantitative measure of error between the predicted and actual angular accelerations. Ongoing research is targeted toward finding a set of quantitative measures necessary for further validation.

ACKNOWLEDGEMENTS

Support for this research was provided in part by a NASA-ASEE Summer Fellowship awarded to the first and second authors. Support was also provided through a NASA-Ames Research Center contract to the second author. The

authors wish to thank Steve Ellis and Dov Adelstein at the HCI division, NASA Ames for their helpful comments and suggestions.

REFERENCES

- [1] K. M. Stanney, R. R. Mourant, and R. S. Kennedy, "Human Factors Issues in Virtual Environments: A Review of the Literature," *Presence*, Vol. 7:4, pp. 327-351, 1998.
- [2] S. R. Ellis, B. D. Adelstein, S. Baumeler, G. J. Jense, and R. H. Jacoby, "Sensor Spatial Distortion, Visual Latency, and Update Rate Effects on 3D Tracking in Virtual Environments," *Proceedings of VR '99*, pp. 218-221, 1999.
- [3] M. Wloka, "Lag in Multiprocessor Virtual Reality," *Presence*, Vol. 4:1, pp. 50-63, 1995.
- [4] M. Friedmann, T. Starner, and A. Pentland, "Device Synchronization Using an Optimal Linear Filter," *Computer Graphics ACM SIGGRAPH*, pp. 57-62, 1992.
- [5] J. Liang, C. Shaw, and M. Green, "On Temporal-Spatial Realism in the Virtual Reality Environment," *Proceedings of the 1991 User Interface Software Technology*, pp. 19-25, 1991.
- [6] J. Y. Jung, B. D. Adelstein, and S. R. Ellis, "Predictive Compensator Optimization for Head Tracking Lag in Virtual Environments," *Proceedings of IMAGE 2000 Conference*, 2000.
- [7] H. Huang and C. Chen, "Development of a Myoelectric Discrimination System for a Multi-degree Prosthetic Head," *Proceedings of 1999 IEEE International Conference on Robotics and Automation*, pp. 2392-2397, 1999.
- [8] J. L. Elman, "Finding Structure in Time," *Cognitive Science*, Vol. 14, pp. 179-211, 1990.
- [9] C. Charalambous, "Conjugate-gradient algorithm for efficient training of artificial neural networks," *IEEE Proceedings*, Vol. 39 (3), pp. 301-310, 1992.