Neural Network 3D Body Pose Tracking and Prediction for Motion-to-Photon Latency Compensation in Distributed Virtual Reality

Sebastian Pohl, Armin Becher, Thomas Grauschopf, and Cristian Axenie(✉)

Electrical Engineering and Computer Science,
Audi Konfuzius-Institut Ingolstadt Lab, Technische Hochschule Ingolstadt,
Esplanade 10, 85049 Ingolstadt, Germany
{sebastian.pohl,armin.becher,thomas.grauschopf}@thi.de,
cristian.axenie@audi-konfuzius-institut-ingolstadt.de

Abstract. Distributed Virtual Reality (DVR) systems enable geographically dispersed users to interact in a shared virtual environment. The realism of the interaction is crucial to increase the feeling of co-presence. Latency, produced either by hard- or software components of DVR applications, impedes reaching high realism levels of the DVR experience. For example, the time delay between the user's motion and the corresponding display rendering of the DVR system might lead to adverse effects such as a reduced sense of presence or motion sickness. One way of minimizing the latency is to predict user's motion and thus compensate for the inherent latency in the system. In order to address this problem, we propose a neural network 3D pose tracking and prediction system with latency guarantees for end-to-end avatar reconstruction. We evaluate and compare our system against multiple traditional methods and provide a thorough analysis on real-world human motion data.

Keywords: Neural networks · Distributed VR systems ·
3D body tracking · Timeseries prediction · Motion-to-photon latency

1 Introduction

Since high-quality consumer-level VR headsets and systems have become commodity there has been a proliferation of novel immersive and interactive VR applications. Although these systems are lightweight, high-resolution and high frame-rate, a remaining obstacle is how to get the user to feel truly immersed in the experience, especially in collaborative scenarios.

In a VR system, when a motion occurs, the motion detection unit samples the orientation data for the view generation. After the motion detection, the visual processing unit renders a 3D image. Finally, the rendered image is outputted to a display corresponding to the head orientation of the user. As these steps

I. V. Tetko et al. (Eds.): ICANN 2019, LNCS 11729, pp. 429–442, 2019.
https://doi.org/10.1007/978-3-030-30508-6_35

take time, the delay results in latency. In this case, the image does not exactly correspond to the actual head orientation of the user, due to the mismatch between visual and vestibular systems, thereby causing the user to experience motion sickness [20].

Motion-to-photon latency, also known as the end-to-end latency, is the delay between the movement of the user's head and the change of the display of VR device reflecting the user's movement. As soon as the user's head moves, the VR scenery should match the movement. The more delay between these two actions, the more unrealistic the VR world seems. To make the VR world realistic, VR systems need low latency of $\leq$20 ms [11] and even really low latency of $\leq$7 ms [14]. This values should hold even if two or more users share the same virtual environment and remote latencies occur between different worldwide VR locations [19,23].

Conventional motion-to-photon latency measurement methods can measure the latency only at the beginning of the physical motion. On the other hand, there are more advanced methods can measure the latency in real-time at every sample [6]. Our previous study [4] supports this idea and estimates that the underlying latency can in theory completely compensated if the system benefits from a precise prediction of position and orientation of the user in 3D. In practice, however, this is not easily achievable because human movements are non-deterministic [14].

The contribution of this paper is a neural network based body pose 3D tracking and prediction system for capturing of avatars with: minimal hardware requirements, an efficient learning system and fast processing time. Correlated with the known latency thresholds, [4,11], the proposed system is able to predict the position of the HMD and the two VR hand controllers for continuous time-based (i.e. timeseries) motion representation subsequently fed to VIRTOOAIR [5], in order to learn the inverse kinematics of the upper-body, global rotation and position of a user inside VR.

2 Related Work

Due to considerable practical relevance in DVR, there is renewed interest in 3D pose tracking and prediction, with impressive preliminary results [17]. Similar works use all the known user skeleton joints. With the kinematic information from the joints, neural networks can predict and generate human motion sequences [18,24]. Yet, in order to be able to track all the joints of the body, one needs expensive motion capture systems. They are accurate for extracting 3D full-body models [9]. However they are very expensive, need long acquisition time (up to minutes), require extensive computing resources and/or take several minutes of processing. Some methods have been proposed to acquire dynamic human models [8] and obtained impressive results but using 106 synchronized cameras and several minutes processing per frame.

On the other end of the spectrum, [13] proposed a method for monocular videos where the body shape can be manipulated in videos. Such systems

required expert human intervention, whereas our aim is a learning system capable to exploit the motion data correlations and extract the underlying kinematic temporal aspects within the motion of the HMD and the hand VR controllers autonomously. The common commercially available VR devices, such as the HTC Vive or the Oculus Rift, cannot provide precise external tracking. These devices provide only the tracking information for the head-mounted display, via which the head position and orientation is calculated, and the data from the two hand controllers. Closer to our approach and taking advantage of parametric models some methods have been proposed to estimate the 3D body from a single picture [25] or from shape variations in range scans [3]. These approaches can provide exciting results and a deformable body model, but they require additional hardware (e.g. range scanners), data representations and complex models (e.g. model-based body-aware image warping). Instead, we look at a learning based approach which runs in a fully automatic manner after training.

Finally, in contrast to other works, our aim is to compensate for latency using the data that VR devices offer. A latency compensation system must be highly real-time capable. In order to benefit from latency compensation, the introduced latency of the predictor should not be larger than the horizon of the prediction. Yet the time the predictive model needs to calculate the next values, depends on the model itself, as we will see later in the experiments section.

3 System Setup and Description

The neural network tracking and prediction system proposed in the paper is a component of VIRTOOAIR: VIrtual Reality TOOlbox for Avatar Intelligent Reconstruction[1] processing pipeline [5]. This is a novel, inexpensive approach to achieve high-fidelity multimodal motion capturing and avatar representation in VR. By fusing an inverse kinematics learning module for precise upper-body motion reconstruction, with single RGB camera input for lower-body estimation the system obtains a rich representation of user's motion. The learning capabilities allow natural pose regression with cheap and affordable marker-less motion capturing hardware.

4 Experiments and Discussion

4.1 Technical Setup

For all the experiments we acquired data from the HTC Vive system [2], that contains an HMD and two hand controllers data, using OpenVR API [1]. The training dataset was further extended with data from Human 3.6M [12]. This online database is composed of 3.6 Million accurate 3D Human poses, acquired by recording the performance of 5 female and 6 male subjects, under 4 different

[1] An overview on the VIRTOOAIR framework is available at: https://audi-konfuzius-institut-ingolstadt.de/category/akii-microlab/current-projects.

viewpoints, for training realistic human sensing systems for pose estimation. The HTC Vive system captures for both the hand controller and the HMD speed and acceleration data via internal sensors, as well position and rotation data via an inside-out tracking system [21]. The implementation of the analysis, predictor models and neural networks was done in Python with Scikit-Learn and Keras API [7]. Tensorflow was used as back-end, which together with CuDNN dispatched the calculations of the networks to an nVidia GTX-1080Ti GPU.

4.2 Data Representation and Pre-processing

The data available from the HTC Vive is in form of a 3×4 transformation matrix T. This matrix is composed of a 3×3 rotation matrix R and the position vector $\boldsymbol{p}$ in 3D space.

$$T = \left[R\,\boldsymbol{p}\right] = \begin{bmatrix} r_{00} & r_{01} & r_{02} & x \\ r_{10} & r_{11} & r_{12} & y \\ r_{20} & r_{21} & r_{22} & z \end{bmatrix}$$

From this matrix we can calculate the Euler angles, but one of the challenges with Euler rotation angles in R^3 space is that they are not unique and suffer from incontinuities. In order to overcome this limitation we use quaternions. A quaternion can be used to represent unambiguously 3D-rotations as a point on a hypersphere in a R^4 space. The quaternion representation $q = (w, x, y, z)$ or $q = w + ix + jy + kz$ is more informative offering a continuous representation, where w, x, y and z are a system of four scalars and i, j and k are three right versors, respectively. In such a representation one needs to consider that human

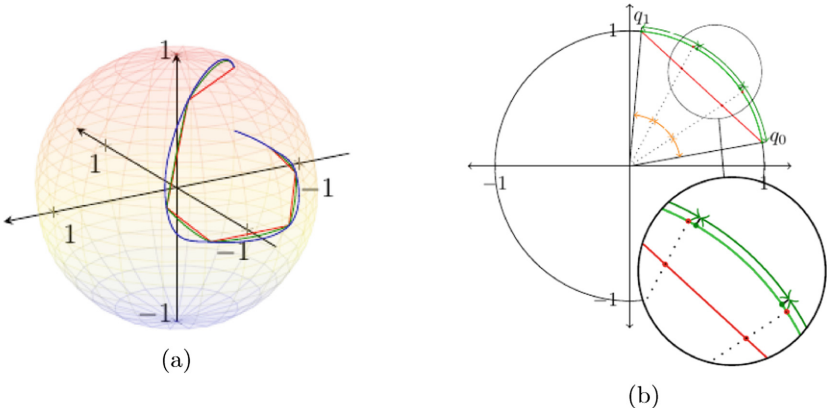

Fig. 1. Visual representation of quaternions on spheres after interpolation: (a) Visual representation of interpolation using *linear interpolation* (red), the *slerp* method (green) and the *squad* (blue), (b) Comparison between linear interpolation (red) and the slerp method (green) between points q0 and q1. (Color figure online)

motion data includes inherent information about the speed and acceleration of a body. Interpolating the data in a realistic form assumes that abrupt changes in the speed, or the acceleration, need to be avoided. In order to achieve this, we explored multiple interpolation methods, such as the linear interpolation, Spherical Linear Interpolation (slerp) and Spherical Spline Quaternion Interpolation (squad), as shown in Fig. 1. Such a step is highly relevant in order to maintain in the training data of our model those subtle changes and informative variance of the motion parameters encoded in the quaternions. As previously mentioned, the data acquired from the tracking system is in the form of time indexed sequences, or timeseries. Choosing quaternions to represent rotations, we performed an initial timeseries correlation analysis to check if there is any interesting effect in the user's head and hands motion that we can exploit with our neural network system. In the correlation matrix in Fig. 2 one can observe that with respect to the absolute positions (Fig. 2, left panel) there is a correlation only between the hand and head positions, but not between the axes themselves. The only correlation visible is the correlation in the first derivative (i.e. the speed) (Fig. 2, right panel). Using the input from the correlation analysis, Fig. 3, we investigated the temporal depth (i.e. lag) to which the position components from both head and hands are correlated, in order to define the prediction horizon in the motion-to-photon latency compensation. We observed that for the position, the autocorrelation of the y position (red) approaches zero but stays positive despite the large lag, whereas the x axis (blue) and z axis (green) become anticorrelated.

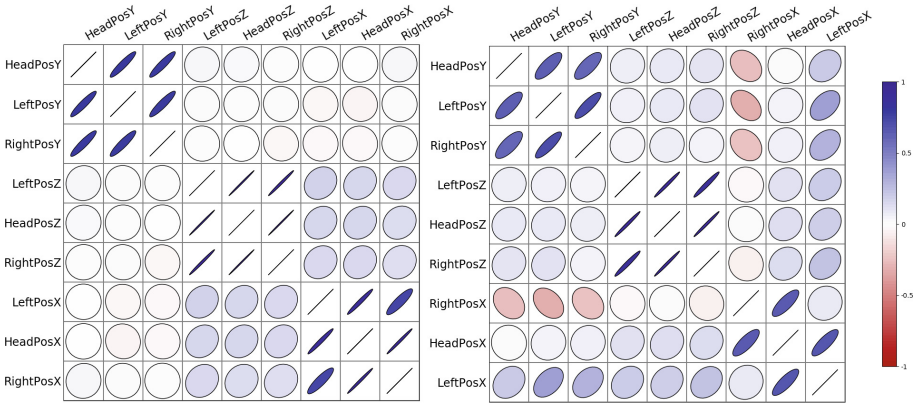

Fig. 2. Motion components correlation analysis. Blue codes for degree of correlation, whereas red anti-correlation of the various 3D motion components. The size of each matrix entry is proportional to the covariance of considered pair of motion components. Left panel: correlation w.r.t. left hand coordinate system; Right panel: correlation w.r.t. right hand coordinate system. (Color figure online)

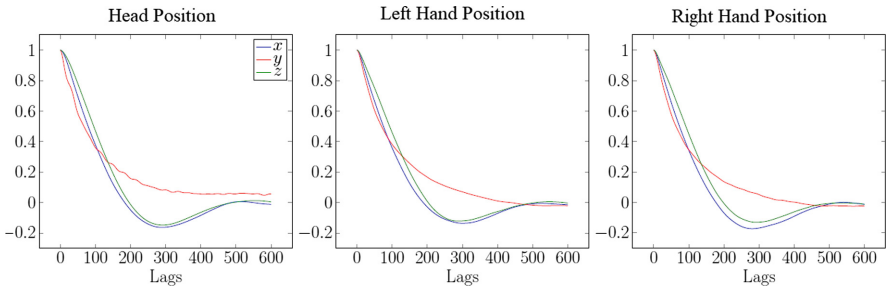

Fig. 3. Position autocorrelation analysis. (Color figure online)

4.3 Neural Network Tracking Model

Human motion prediction aims to understand behaviors of a subject on the observed sequences and to generate future body poses. Most of the models are focused on image data as input instead of other source of data (e.g. inertial, inside-out tracking). Recurrent Neural Networks (RNN) have shown exciting results in: learning 3D motion temporal dynamics in a sequence [10,24], to automatically learn the image-dependent structural and temporal constraints using long short-term memory (LSTM) units [16], and up to LSTM-based deep learning architectures for 3D depth reconstruction by learning the intrinsic joint interdependency [15].

In our work, aiming at compensating for motion-to-photon latency, we deliberately chose a neural model capable of fast and precise prediction using data from the HTC Vive system. Nonlinear autoregressive networks (NAR), Fig. 4, represent a class of recurrent networks that are suitable for time series calculations. The input of such a neural network is a sub-sequence of observations of a time series. The number of observations depends on the chosen order p. The network thus learns to calculate the unknown function $f(\cdot)$ from the observations x_n to x_{n-p}. Hence, the corresponding function can be described as

$$\hat{x}_{n+1} = f(x_n, x_{n-1}, \cdots, x_{n-p}). \tag{1}$$

The network receives as input a certain number of lags of a time series and learns a function to estimate the respective next point of this timeseries. The computation can be implemented as open-loop as well as closed-loop. In the open-loop, the real value of x_{n+1} is used next, and the remaining values x_{n-1} to x_{n-p} are delayed by one input neuron. The oldest value x_{n-p} drops out and will not be used any further. Since there are no explicit feedback connections within the network, the horizon of the timeseries over which dependencies can be learned is limited to the chosen order of autoregression. The closed-loop approach is analogous to the open-loop, with the only difference being that instead of the true value for x_{n+1}, the previously estimated value $\hat{x}_{n+1}$ is used. Depending on how far the forecast should reach into the future, this procedure can be repeated

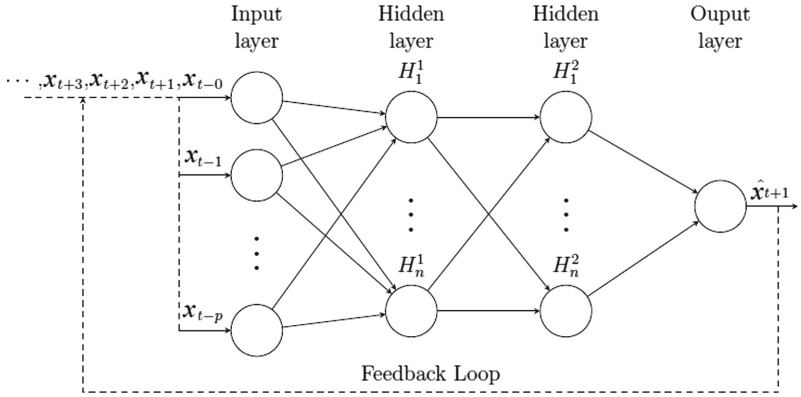

Fig. 4. Neural network model architecture.

as often as desired. In such a network the core assumption is that subsequent measurements are intrinsically spatio-temporally correlated. In order to evaluate the performance of the proposed neural model the prediction of the next point $t+1$ is done using the open-loop prediction with the real data as input. Further predictions with for $t+k$ with $k>1$ are made via closed-loop operation. Note that in some of the literature, the terms "pose prediction" and "pose estimation" are used interchangeably, both referring to the task of estimating a pose (usually skeleton-based joint locations). In this work, we use the term "prediction" to refer to the specific task of predicting/forecasting 3D user pose at time $t+1$ in a sequence, assuming the 3D poses were already estimated at time t.

4.4 Evaluation

In order to provide a complete and fair analysis of our proposed neural network model, we evaluated state-of-the models for timeseries prediction (i.e. Vector Autoregressive (VAR) Models) as well as machine learning regression techniques (i.e. Ridge Regression). For evaluation, we explored multiple multivariate time-series metrics due to the nature of our input data and found that the Mean Absolute Scaled Error (MASE) would be a good candidate. From the basic MASE formulation, because there is no restriction on the mean, we divide the prediction of a model by the mean, allowing us to also extract more statistics (e.g. median)

$$q_j = \frac{e_j}{\frac{1}{T-1} \cdot \sum_{t=2}^{T} |y_t - y_{t-1}|} \tag{2}$$

so

$$MASE = \frac{1}{N} \cdot \sum_{j=1}^{N} q_j. \tag{3}$$

This metric is very easy to interpret. As Fig. 5 shows, when applied to a model in comparison to the naive forecast (or another benchmark model), for

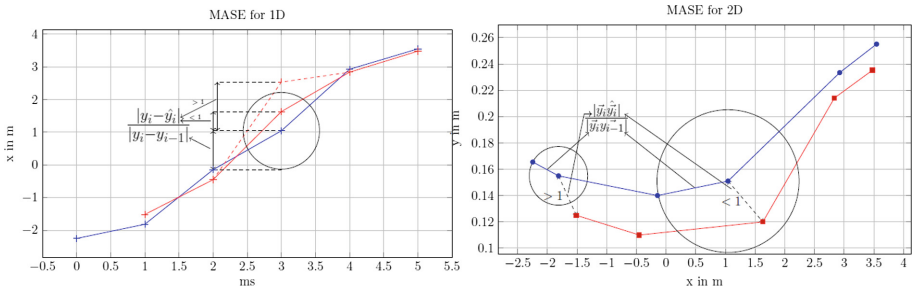

Fig. 5. Evaluation of MASE metric. Blue curve is the true motion, whereas red is the prediction of the system. The equations describe the way the metric is constructed in each dimension. (Color figure online)

values < 1 the model is better, for values be > 1 the model is worse. Since the MASE metric is designed for univariate timeseries, we update the formulation, instead of the absolute deviation between two points of the time series we will use the Euclidean distance

$$|y\hat{y}| = EE = \sqrt{\sum_{i=1}^{d}(y_i - \hat{y}_i)^2} \Rightarrow MEE = \frac{1}{T}\sum_{t=0}^{T}\sqrt{\sum_{i=1}^{d}(y_{i,t} - \hat{y}_{i,t})^2}. \quad (4)$$

For a univariate time series, this procedure is equivalent to normal calculation of the MASE, since the Euclidean distance in the one-dimensional space is equal to the absolute distance. Finally, we formulate MESE (Mean Euclidean Scaled Error) as a second metric that looks at position changes in time

$$MESE = \frac{1}{T} \cdot \sum_{t=1}^{T} \frac{|y_t\hat{y}_t|}{\frac{1}{N-1}\sum_{n=2}^{N}|y_n y_{n-1}|}. \quad (5)$$

In order to guide the reader through the evaluation in Fig. 3 we mention that, in practice, a deviation of almost 2 cm is acceptable for DVR setups. For rotation prediction, we convert quaternions in rotation angles through a nonlinear transformation (assuming unitary quaternions)

$$cos(\theta) = \hat{q} \cdot q, \theta = arccos(\hat{q} \cdot q). \quad (6)$$

The first evaluated model is the VAR model, a multivariate autoregressive model for which the prediction is given by

$$\hat{\boldsymbol{y}}_{t+1} = f(y_{1,t}, \cdots, y_{k,t}, y_{1,t-1}, \cdots, y_{k,t-1}, y_{1,t-p}, \cdots, y_{k,t-p}). \quad (7)$$

For this relatively simple approach we analyse, in Table 1, the MEE, the MESE, and the angular deviation of the quaternions for VAR models with predictions of up to 25 ms in 5 ms intervals. The model input are the last 6 lags of the respective position axes or the four scalar values of the quaternions of the

Table 1. Evaluation of VAR prediction.

VAR						
	Metric	5 ms	10 ms	15 ms	20 ms	25 ms
Head position	MEE	0.0005	0.0070	0.0138	0.0207	0.0274
	MESE	0.0937	1.0230	2.0144	3.0043	3.9882
Left Hand Position	MEE	0.0011	0.0101	0.0195	0.0290	0.0383
	MESE	0.1224	1.0541	2.0420	3.0235	3.9932
Right Hand Position	MEE	0.0015	0.0115	0.0221	0.0328	0.0433
	MESE	0.1489	1.0694	2.0517	3.0272	3.9940

hands and of the head. This value was chosen as significant (i.e. cut-off lag) in the Auto-Correlation Function plot (ACF), Fig. 3, considering 95% confidence interval. As one can see, there is an error in the estimate of the position of the head. After 5 ms the error is half a millimeter and is only after 5 ms that reaches about an inch. The error of the hand positions already exceeds one centimeter after 10 ms and are at over one millimeter at 5 ms. If the input vector consists of several variables, it can lead to multi-collinearity between the descriptive variables. This can lead to a poor estimate of the model parameters [22]. Ridge Regression solves this problem by extending the least squares method using a $L2$ regularization term

$$\sum_{i=1}^{n}(y_i - x_{ij}\beta_j)^2 + \lambda\|\beta\|_2^2. \tag{8}$$

In multivariate cases the estimation of the β parameters is performed by

$$\hat{\beta} = (X^T X + \lambda I)^{-1} X^T Y. \tag{9}$$

Similarly to the VAR model, we evaluate the Ridge Regression model, as shown in Table 2. As we can observe, we have nearly identical to the errors of the VAR model. The position of the head is with a mean error of half a millimeter

Table 2. Evaluation of Ridge Regression prediction.

Ridge Regressor						
	Metric	5 ms	10 ms	15 ms	20 ms	25 ms
Head Position	MEE	0.0005	0.0070	0.0138	0.0206	0.0274
	MESE	0.0939	1.0237	2.0142	3.0037	3.9881
Left Hand Position	MEE	0.0011	0.0100	0.0195	0.0290	0.0383
	MESE	0.1229	1.0542	2.0421	3.0232	3.9926
Right Hand Position	MEE	0.0014	0.0115	0.0221	0.0328	0.0433
	MESE	0.1479	1.0698	2.0506	3.0272	3.9934

Table 3. Evaluation of Neural Network prediction.

Neural Network						
	Metric	5 ms	10 ms	15 ms	20 ms	25 ms
Head Position	MEE	0.0064	0.0073	0.0088	0.0107	0.0135
	MESE	1.1288	1.3052	1.5890	1.9593	2.4957
Left Hand Position	MEE	0.009016	0.008784	0.009181	0.011064	0.014852
	MESE	0.944028	0.939354	1.008508	1.233179	1.650189
Right Hand Position	MEE	0.0108	0.0105	0.0110	0.0131	0.0175
	MESE	0.9588	0.9672	1.0405	1.2617	1.6679

better prediction. From 15 ms the error is increasing over one centimeter. The hand positions deviate from 10 ms by over one centimeter. We complete our analysis with the evaluation of our neural network model performance. Similarly to state-of-the-art model, we analyse, in Table 3, the MEE and MASE as well as the statistics of the prediction. This will emphasize the advantages of the proposed system. Similarly to the Ridge Regression, the estimated position of the head is better than that of the hands. The error of the head position is after 5 ms at something more than 0.5 mm. At the same prediction horizon, the hand positions already deviate about an inch. Nevertheless, the MESE metric for the hands is lower than that of the head. This is explained by the fact that the hands move much more than the head does. Therefore, the counter for the MESE metric is larger for the hands than for the head, whereby the total value decreases. In terms of a thorough statistical evaluation of the prediction at various horizon sizes, the neural network model is excelling at horizon levels between 5 and 10 ms, as shown in Table 4. These values are consistent with the interval of interest in motion-to-photon compensation, [4]. This work shows that the prediction of human movements for DVR is difficult. The best prediction was achieved using a neural network model (i.e. NAR) with a global error that lies however after 10 ms in over 25% of the predictions, at over 1 cm deviation. This is still under the guarantees of compensating the motion-to-photon latency and achieved with a simple model (i.e. NAR) compared to other approaches employing LSTMs or other such computationally expensive methods. The used combination of data representation and the neural network model provide a good match for the problem of predicting 3D pose in VR. The errors of the neural network model are on average lower for the prediction for the time steps $t+1$, but not over the entire range of prediction horizons. Table 6 shows the coefficients of variation ($\frac{std}{mean}$) for the head position estimated by the neural network and the Ridge Regression estimate. Here it can be seen that for the first time steps, the variation relative to the mean value for the ridge regression is almost twice as high as that of the neural network. It can be seen from the Tables 3 and 4 that, despite the higher mean error, the maximum error in the neural network model is lower than that of a ridge regression. For prediction horizons larger

Table 4. Evaluation of Neural Network prediction.

		Std	Min	25%	50%	75%	Max
Head	5 ms	0.006138	0.000026	0.001614	0.004255	0.009721	0.052726
	10 ms	0.006847	0.000044	0.002014	0.005024	0.010839	0.064883
	15 ms	0.008053	0.000040	0.002647	0.006258	0.012710	0.086756
	20 ms	0.009730	0.000082	0.003433	0.007806	0.015189	0.123706
	25 ms	0.012079	0.000062	0.004575	0.010032	0.018925	0.162462
Right Hand	5 ms	0.010157	0.000022	0.003026	0.008582	0.015398	0.261596
	10 ms	0.011207	0.000077	0.003373	0.008108	0.014211	0.430261
	15 ms	0.013530	0.000108	0.003997	0.007916	0.013682	0.659150
	20 ms	0.016923	0.000054	0.005132	0.009362	0.015498	0.821064
	25 ms	0.021201	0.000067	0.007047	0.012942	0.021082	1.022083
Left Hand	5 ms	0.008447	0.000024	0.002343	0.006984	0.012944	0.148632
	10 ms	0.008700	0.000023	0.002750	0.006589	0.011913	0.267453
	15 ms	0.009972	0.000074	0.003412	0.006587	0.011584	0.362364
	20 ms	0.012425	0.000058	0.004415	0.007987	0.013395	0.451241
	25 ms	0.015777	0.000144	0.006147	0.011124	0.018236	0.549077

than 15 ms, the errors of the neural network model are consistently lower than is the case with a ridge regression. With an upper quantile limit of 1.89 mm, the errors after 25 ms are still lower than the errors of the Ridge Regression after 15 ms, as shown in Table 5. As previously mentioned, a deviation of almost 2 cm is acceptable for the DVR setups we are considering.

Table 5. Evaluation of Ridge Regression prediction.

		Std	Min	25%	50%	75%	Max
Head	5 ms	0.000911	8.613443e-07	0.000219	0.000379	0.000636	0.091600
	10 ms	0.006241	7.789306e-06	0.001705	0.004993	0.011511	0.124571
	15 ms	0.012349	6.221113e-06	0.003302	0.009812	0.022824	0.171270
	20 ms	0.018467	2.252768e-05	0.004897	0.014620	0.034124	0.189392
	25 ms	0.024569	2.397779e-05	0.000473	0.019391	0.045408	0.251757
Right Hand	5 ms	0.002222	0.000003	0.000333	0.000651	0.001210	0.106050
	10 ms	0.009895	0.000007	0.002373	0.007526	0.014565	0.163459
	15 ms	0.019207	0.000015	0.004519	0.014631	0.028474	0.315861
	20 ms	0.028501	0.000013	0.006642	0.021697	0.042239	0.462435
	25 ms	0.037669	0.000024	0.008729	0.028683	0.055882	0.G03335
Left Hand	5 ms	0.002821	0.000004	0.000444	0.000827	0.001539	0.140264
	10 ms	0.010495	0.000011	0.003057	0.009175	0.016775	0.175692
	15 ms	0.020177	0.000009	0.005757	0.017663	0.032617	0.266510
	20 ms	0.029895	0.000024	0.008430	0.026088	0.048387	0.378546
	25 ms	0.039513	0.000017	0.011057	0.034422	0.064057	0.477469

Table 6. Variation coefficient of Ridge Regression vs. Neural Network.

Variation coefficient (sd/mean)					
	5 ms	10 ms	15 ms	20 ms	25 ms
NN	0.9590	0.938	0.909	0.909	0.895
Ride	1.822	0.891	0.895	0.892	0.896

The thorough evaluation and comparative analysis provided in this section emphasize some advantages the proposed neural network model has in predicting head and hand position and orientation under the constraint of motion-to-photon latency. This is also visible in the quatitative analysis of variation coefficients in Table 6.

Such a system can bring benefits in actively compensating for latency and complete the end-to-end pipeline towards a realistic VR avatar reconstruction in DVR systems such as the one proposed in VIRTOOAIR[2].

5 Conclusions

DVR is still in infancy but will definitely require digital alter egos of the users' physical selves, virtual replicas termed avatars. Such embodied interfaces to the artificially generated environments provide a means of direct interaction with the environments based on the simulation of physical properties. Motion tracking and prediction is a key ingredient for a realistic immersion. Yet, motion-to-photon latency must be compensated taken into account the variability of human motion. The proposed neural network system exploits the underlying correlations in the upper body kinematics and provides latency guarantees for a natural VR experience. As such avatars are our proxies in the DVR, they are the direct extension of ourselves into the virtual domain, hence their digital representations should be tightly bound to our motion, our self-perception, and, why not, our personality.

References

1. Openvr sdk (2015). https://github.com/ValveSoftware/openvr
2. Vive vr system (2018). https://www.vive.com/us/product/vive-virtual-reality-system/
3. Allen, B., Curless, B., Popović, Z., Hertzmann, A.: Learning a correlated model of identity and pose-dependent body shape variation for real-time synthesis. In: Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA 2006, pp. 147–156. Eurographics Association, Aire-la-Ville (2006). http://dl.acm.org/citation.cfm?id=1218064.1218084

[2] Source code available at: https://gitlab.com/akii-microlab/virtooair.

4. Becher, A., Angerer, J., Grauschopf, T.: Novel approach to measure motion-to-photon and mouth-to-ear latency in distributed virtual reality systems. In: GI VR/AR WORKSHOP 2018 (2018)
5. Becher A., Axenie C., Grauschopf, T.: VIRTOOAIR: virtual reality toolbox for avatar intelligent reconstruction. In: Multimodal Virtual and Augmented Reality Workshop (MVAR) at 2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR) (2018)
6. Choi, S.W., Lee, S., Seo, M.W., Kang, S.J.: Time sequential motion-to-photon latency measurement system for virtual reality head-mounted displays. Electronics **7**(9), 171 (2018)
7. Chollet, F., et al.: Keras (2015). https://github.com/fchollet/keras
8. Collet, A., et al.: High-quality streamable free-viewpoint video. ACM Trans. Graph. **34**(4), 69:1–69:13 (2015). https://doi.org/10.1145/2766945
9. Daanen, H., Ter Haar, F.: Review. Displays **34**(4), 270–275 (2013)
10. Du, X., Vasudevan, R., Johnson-Roberson, M.: Bio-LSTM: a biomechanically inspired recurrent neural network for 3-d pedestrian pose and gait prediction. IEEE Robot. Autom. Lett. **4**(2), 1501–1508 (2019)
11. Elbamby, M.S., Perfecto, C., Bennis, M., Doppler, K.: Toward low-latency and ultra-reliable virtual reality. IEEE Netw. **32**(2), 78–84 (2018)
12. Ionescu, C., Papava, D., Olaru, V., Sminchisescu, C.: Human3.6m: large scale datasets and predictive methods for 3D human sensing in natural environments. IEEE Trans. Pattern Anal. Mach. Intell. **36**(7), 1325–1339 (2014)
13. Jain, A., Thormählen, T., Seidel, H.P., Theobalt, C.: Moviereshape: tracking and reshaping of humans in videos. In: ACM SIGGRAPH Asia 2010 Papers, SIGGRAPH ASIA 2010, pp. 148:1–148:10. ACM, New York (2010). https://doi.org/10.1145/1866158.1866174
14. Jerald, J., Whitton, M., Brooks Jr., F.P.: Scene-motion thresholds during head yaw for immersive virtual environments. ACM Trans. Appl. Percept. **9**(1), 4:1–4:23 (2012)
15. Lee, K., Lee, I., Lee, S.: Propagating LSTM: 3D pose estimation based on joint interdependency. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11211, pp. 123–141. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_8
16. Lin, M., Lin, L., Liang, X., Wang, K., Cheng, H.: Recurrent 3D pose sequence machines. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
17. Malleson, C., et al.: Rapid one-shot acquisition of dynamic VR avatars. In: 2017 IEEE Virtual Reality (VR), pp. 131–140 (2017)
18. Martinez, J., Black, M.J., Romero, J.: On human motion prediction using recurrent neural networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4674–4683 (2017)
19. Meehan, M., Razzaque, S., Whitton, M.C., Brooks, F.P.: Effect of latency on presence in stressful virtual environments. In: IEEE Virtual Reality 2003, pp. 141–148, 22–26 March 2003
20. Munafo, J., Diedrick, M., Stoffregen, T.A.: The virtual reality head-mounted display oculus rift induces motion sickness and is sexist in its effects. Exp. Brain Res. **235**(3), 889–901 (2017)
21. Niehorster, D.C., Li, L., Lappe, M.: The accuracy and precision of position and orientation tracking in the HTC vive virtual reality system for scientific research. i-Perception (2017)

22. Shih, S., Shih, W.: Application of ridge regression analysis to water resources studies. J. Hydrol. **40**(1), 165–174 (1979)
23. St. Pierre, M.E., Banerjee, S., Hoover, A.W., Muth, E.R.: The effects of 0.2hz varying latency with 20–100ms varying amplitude on simulator sickness in a helmet mounted display. Displays **36**, 1–8 (2015)
24. Tang, Y., Ma, L., Liu, W., Zheng, W.S.: Long-term human motion prediction by modeling motion context and enhancing motion dynamics. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (2018). https://doi.org/10.24963/ijcai.2018/130
25. Zhou, S., Fu, H., Liu, L., Cohen-Or, D., Han, X.: Parametric reshaping of human bodies in images. ACM Trans. Graph. **29**(4), 126:1–126:10 (2010)