Using Artificial Neural Networks to Predict Avatar Movement in Virtual Reality

David Halbhuber
Universität Regensburg
Regensburg, Germany
david.halbhuber@stud.uni-r.de

Jonathan Sasse
Universität Regensburg
Regensburg, Germany
jonathan.sasse@stud.uni-r.de

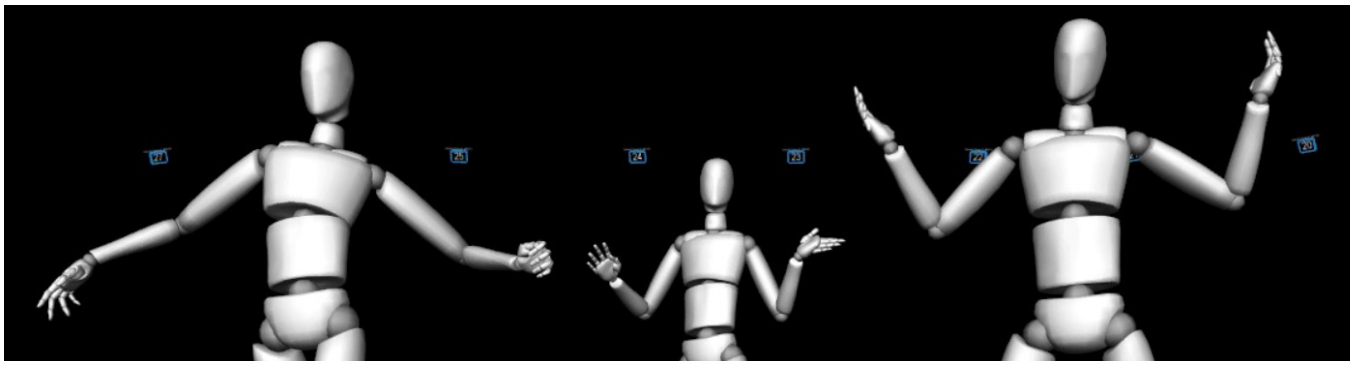


Figure 1: Motion captured avatars showing different postures.

ABSTRACT

In this work we present a system to predict human avatar movement in a virtual reality environment. This prediction is meant to compensate the system latency caused by the motion capturing system and the head mounted device used in the environment, since this latency is believed not only to worsen the presence and the feeling of immersion of the user, but also its performance in completing certain tasks. Firstly, we have developed a system that is able to reliably determine the system latency of the used setup. Secondly, we present two different neural networks, which on the one hand are able to predict the movement 48ms in the future, and on the other hand is able to predict 100ms in the future. Thirdly, we evaluated the proposed system with a user study. It is shown that the used system neither improved nor worsened the presence or the feeling of immersion. Furthermore, no increase in performance was observed. We were not able to reveal a significant effect using either of the presented models. Additional research is needed to further investigate the link between latency in virtual reality systems and the experience a user undergoes.

CCS CONCEPTS

• **Human-centered computing** → **Interaction design**.

KEYWORDS

motion capturing, artificial neural networks, movement prediction, virtual reality

ACM Reference Format:
David Halbhuber and Jonathan Sasse. 2019. Using Artificial Neural Networks to Predict Avatar Movement in Virtual Reality . In *Proceedings of ACM Conference (Conference'19).* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

State-of-the-art motion capturing systems such as NaturalPoint's OptiTrack[22] can be utilized to transfer movement to a computer generated space. This space can be used to integrate movement in several different kinds of computer applications, such as games, movie recording software or any kind of application that aims to mimic human movement as accurate as possible. In a human-centered approach the movement of a human gets transferred to the application, this allows the user to directly interact with the software through natural movement.

One step further in directly integrating natural human movement and behaviour in the above mentioned applications can be obtained by combining state of the art motion capturing devices with head mounted displays (HMD). This devices allow the user to see the generated computer space the so called Virtual Reality (VR). By combining a motion capturing system and a HMD the user is able to see its avatars movement in VR through its own eyes. Such systems have already been widely researched. One example of this is the work of Chua et al.[4] in which the authors propose and evaluate a method to learn Taijiquan, an ancient Chinese form of material arts, using a motion capture system in combination with a HMD. Although this type of system is promising for a wide variety of applications, there are some disadvantages and system weaknesses.

Since many different technical components and systems are connected during operation, a certain technical latency arises in the course of communication between devices. This means that the user using the system is not able to see its movements in real time. If, for example, the user lifts its arm, the system takes some time to register, process and transmit this movement to the HMD, which itself needs some time to display the received data correctly. The processing and transmission time results in a perceivable latency and can ultimately lead to a reduced feeling of presence[21, 24].

The loss of presence can be detrimental to the experience the user encounters while using the system, as this determines whether the user feels integrated or out of place in the computer-generated world[8, 26]. Furthermore a high latency value is able to influence a users overall performance in completing and perceiving tasks as well[18]. Figure 2 shows a generalized illustration of a motion capturing system in combination with a HMD.

This work aims to solve the problems, which arise with a high and perceivable system latency, through a movement prediction mechanism based on artificial deep neural networks (DNN). The presented system is capable of predicting user movement by calculating probabilities of movement based on a prior learned data set. DNNs, and artificial neural networks in general, have already achieved remarkable results in predicting different system states, such as the anticipated traffic volume of a crossroad[13] or the amount of expected rainfall in a certain region[19].Likewise, artificial neural networks are already being used to detect human movements, such as pedestrian movements in road traffic[17, 31, 37].

The following paper is structured as follows. The next section briefly exhibits related work in the fields of latency compensation methods, movement prediction using neural networks and finally immersion in virtual reality Then, we describe our developed system to reliably determine the system latency of the used motion capturing system. Section 4 describes the designed study to gather the needed data for training the used neural network. Then the developed neuronal network is described. Different developed and tested architectures are evaluated. Subsequently, the conducted study to measure impact of our designed system is evaluated. Following this, a section discusses the presented results. This paper closes with a section on discussions and conclusions.

2 RELATED WORK

This section provides a brief overview of existing research in the various adjacent research fields. First of all, we discuss existing possibilities for latency compensation as well as their applicability in our scenario. Subsequently, we present papers that deal with the prediction of human movements through neural networks. The following subsection deals with the measurement and interpretation of immersion in virtual reality environments. This section concludes with a summary and on how the gained knowledge will find use in the presented paper.

Latency compensation

Latency compensation is not a new research field. As early as the year 2000 Wu and Ming demonstrated that latency greatly influences human performance in spatial tasks[41]. The authors developed a system which is able to track and predict the head movement of a human using a HMD. The presented system uses simple linear extrapolation, called Kalman filtering[14], to assume and calculate future head positions. Kalman filtering is based on the assumption that the values to be predicted are describable by a mathematical model such as a motion equation. A similar assumption is made when using a neural network to predict any value, with the exception that usually neural networks utilize a so called activation function. The used activation transform a former linear extrapolation to a non linear regression-like problem. Using the Kalman filtering method the authors were able to evidently show that a human can improve task performance by up to 120 percent when using latency compensation methods. Since neural networks are capable to coup with way more complex problems, using them to predict the movement of the whole body and thus effectively reducing the system latency is a promising approach.

Movement prediction through neural networks

Human movement prediction is an active field of research in many various areas. One extraordinary achievement in this research field lays in the work of Martinez, Black and Romero[20]. In their paper, the authors show that traditional Recurrent Neural Networks are able to outperform the performance of complex state-of-the-art networks, like LSTM-3LR[6]. This superior performance in predicting human motion is achieved through meticulous hyper-parameter tuning, such as increasing the batch size instead of reducing the learning rate, and the use of a sequence-to-sequence approach similar to that used in common translation algorithms. The authors justify this approach with the difficulty to correctly implement noise vectors in a time-dependent data set of human movements. The sequence-to-sequence variant is implemented by two neural networks - an encoder network and a decoder network. The encoder network is trained to generate plausible noise vectors based on the available motion data. The decoder network on the other hand is trained with the extended data set (motion data and noise vectors) for the actual motion prediction. Furthermore, the authors point out that the so-called zero velocity joints have received too little attention in the past. By using a baseline based on zero-velocity joints the authors were able to achieve an even better result in the short-term prediction of human movements. It is to be shown if RNNs in general and the above presented method of baseline building is applicable in this work.

Presence in virtual reality

The term presence refers to the phenomenon of behavior and feeling as if the user were in the virtual world created by computer screens, in case of this work the world created and displayed on the HMD[26]. The presence feeling in VR can be a decisive factor in various factors. In a study with almost 1000 participants, Tussyadiah et al. showed that presence highly increase the general enjoyment of VR applications[38], furthermore the authors concluded, that a higher level of "being there", of presence, over all positively betters VR experiences.

But also VR applications with much more specific use cases don't only profit from a high presence feeling, but presuppose it compellingly. Consequently , Price and Anderson showed how the presence feeling affects the success of VR confrontation therapies[23]. Through the conducted study, Price and Anderson showed that there is a strong correlation between presence and therapy success. Also during the therapy a high presence feeling is decisive, because without the feeling to be actually in the VR, no fear triggering element could be released in many test persons. According to the authors, a confrontation therapy that cannot confront does not promise any real success.

Interestingly, related work also shows how a high presence feeling can help decouple the user from one's own body. The authors Hoffmann et al. showed how a sort of analgesia could be induced in test persons by using High Tec VR technology[16].In their work, the authors show how the pain perception of volunteers can be significantly changed by the use of VR technology. The subjects were tested in three different settings (no VR, Low Tec VR, High Tec VR), in each setting the subject was exposed to a low pain stimulus. Finally, the study concludes that a high presence in VR leads to reduced pain perception in the real world.

Summary

The presented related work suggests a high connection between several, in this work researched, phenomena. Firstly it is depicted that latency and performance correlated. Secondly it is shown that artificial neural networks are already capable of predicting human movement behaviour. Lastly it shows that that presence tremendously influences the outcome and the execution of highly specialized VR applications. Additionally it is shown, how presence is able to influence the users perception, bidirectionally. By combining the above findings, we aim to develop a system which not only increases the presence of VR users by depicting a more accurate presentation of themselves, but also increases the performance in completing certain task through compensating the occurring system latency.

3 LATENCY TEST FRAMEWORK

The latency reducing system presented in this paper consists of two technical subsystems which on the one hand are to be understood as requirements for the movement prediction system to be developed and on the other hand represent the actual movement prediction system itself. Our Latency Test Framework(LTF), the first technical subsystem, is needed to determine the actual latency of the used motion capturing system in combination with the utilize HMD. The movement prediction system on the other hand is used to actually predict movement in the presented virtual environment. To minimize the latency between the users actual movement and the shown view in VR using the developed movement prediction system, it is essential to determine the exact latency first. In order to accomplish an highly accurate measurement, independent of confounding variables and users,the LTF, based on various hardware sensors coupled with a Python interface was developed.

The hardware part of the LTF consists of an Arduino micro controller[1] which is linked to a shock sensor and a light sensitive photo sensor. NaturelPoints OptiTrack motion capturing system is used to capture the motion of an user. The captured motions are sent to a computer running the 3D game engine Unity[36] which passes the motion on to the connected HTCs HMD Vive[11]. On the software side, multiple applications and scripts are used to react to the deflections of the hardware sensors.

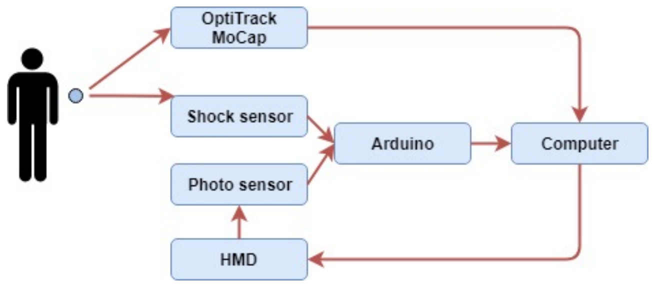


Figure 2: The workflow of the developed Latency Test Framework (LTF). An objects movement gets tracked by the MoCap system. The object also triggers a shock sensor attached to an Arduino micro controller and and virtual trigger zone. Triggering this zone commands the computer to brighten the connected HMD. The change of brightness is received by the attached photo sensor. The overall latency is calculated by subtracting the time the photo sensor got triggered by the time the initial shock was registered.

Figure 2 depicts the workflow of the LTF. The process gets initiated by an user dropping an object, in this case a small blue ball. While the object and its fall are tracked by the motion capturing system, the vibration sensor attached to a wooden panel is triggered when hit. The Arduino connected to the vibration sensor is informed about the event and writes a timestamp in microseconds, which is then passed to a external computer. A Python script running on this computer receives and sets the timestamp of the vibration as starting value for the latency calculation.

In the meantime the OptiTrack system receives and processes the movement of the object and transfers the information onto the central computer. The movement data is being processed and rendered on the HMD using the Unity 3D game engine. Additionally the Unity application registers the objects collision with the wooden pane. Upon collision Unity starts a function to light up the rendered view which is dark by default.

The brightening is registered by the photo sensor detecting the change in light intensity. As soon as the photo sensor is triggered a signal gets passed to the Arduino, which then passes the second timestamp to the central computer.

By comparing the two timestamps determined by the vibration and the photo sensor, it is possible to accurately calculate the time needed for the system to register an object and display it on the HMD. Since the timestamps are recorded directly on the Arduino and only sent to the connected computer there is no additional latency generated by the communication between computer and Arduino. Using our system we determined a mean latency of 55ms with a standard derivation of 7 ms (n=120).

Our presented system is highly accurate, but there a some minor shortcomings in the used methodology. Firstly, it was not possible to reliably determine the response time (also known as activation time) of the used sensor modules manually. According to the sensors manufacturer, both have a response time of less than 2 micro seconds(us). The second weak point represents the connection between the sensors used and the Arduino micro controller. The time stamps required for the LTF are not set until the signal from the sensors reaches the Arduino. The time required for the transmission of the signal is not taken into account. Since the cables used are only very short, a loss-free transmission can be assumed[39]. The transmission speed in a lossless scenario is given by the general solution to Telegrapher's line transmission equations[25]:

$$u = \frac{1}{\sqrt{LC}} \tag{1}$$

The result, u, of the above formula is known as the propagation speed (often also phase speed) of electromagnetic waves. The value u depends on L, the inductance of the conductor, as well as on C, the capacitance of the conductor. Both values are dependent on the length of the conductor, as the conductors used in the LTF are extremely short, the term LC closes to near zero values. This results in the signal transmission from sensors to the Arduino close to the maximum transmission speed of electromagnetic waves, the speed of light. The time required for the transmission is so short, in fact less than 1 microsecond, that it can be neglected when using the LTF. The last weak point of the system is the used Arduino itself. It cannot be reliably determined how long the Arduino needs to process the received data. In order to keep the processing time as short as possible, the code used had to be maximised in terms of efficiency. The theoretical minimum processing time of the Arduino is 4 microseconds, but this is due to the resolution of the microsecond function (used to track latency and processing time) of the Arduino which only works with a resolution of 4 microseconds. This means that a processing time of less than 4 microseconds can only be represented by the Arduino as 4 microseconds.

In summary, the expected error due to all weak points amounts to a maximum of 7 microseconds. An error of this magnitude can be neglected in this paper without negative consequences. The next section describes our data acquisition study.

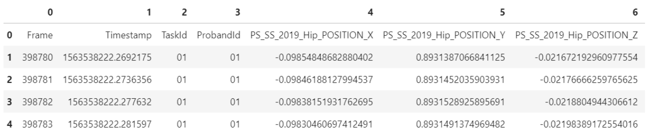
4 DATA ACQUISITION

A preliminary data acquisition study was conducted to gather the data necessary for training the presented artificial neural network. For this purpose we recruited 20 volunteers (11 m, 9 f) with an mean age of 24,4 years (SD 4,0), which were recorded while participating. A few requirements and criteria for the participants were required in order to capture usable data. Subjects where those were not met were excluded from the study and their data deleted, regardless of whether specifications failed prior or during the study.

Participants should not bet compromised in their movement due to illness, physical disability or high age. Minors did not participate in the study because of the requirement for signing a consent form for using their data which would cause an administrative burden. Aside from these requirements, there were no prior experience needed. At the start of the data acquisition study participants had to sign a consent form to accept the collection of personal data as part of the used questionnaire and motion capturing. Afterwards they were introduced to the temporal sequence, their involvement and a few basic guidelines, such as avoiding masking markers during recording. Each task began and ended with the participant posing in T-position.

For the study the participants had to execute 17 varying short tasks, like waving with switching hands or mime a sport they love. The tasks were designed to detail the entire range of motion of the human body, with as many variations as possible. In order to increase the individualization and the later generalization capability of the neuronal network, only the scope of some tasks was given by the experimenter. How the tasks were subsequently executed by the test person could be freely decided by the test person themselves. There were 4 different task sequences in a randomized order. Each individual was encouraged to execute each task for about 30 seconds while changing directions, speed or using another limb if achievable. For capturing and processing the data the group of participants were wearing the HMD and a motion capturing suit coupled to the OptiTrack system. To record the data, we use our self-developed client. The client catches the bit-stream of the Optitrack system, unpacks the packets and saves their contents in a CSV file. Each packet contains one frame of the 240Hz stream of the motion capturing system. A frame in turn describes the corresponding skeleton created using OptiTrack. The skeleton created by OptiTrack consists of 51 different bones. For each bone, OptiTrack provides information on the current position and rotation. For the correct representation of the skeleton in third party applications, such as Unity, only the position of the hip as well as the rotational movement of all bones is necessary. The remaining position information was therefore not recorded by our client. In sum, the float values to be recorded for each frame amounted to a total of 207 (X-position hip, Y-position hip, Z-position hip, as well as X-rotation, Y-rotation, Z-rotation, W-rotation for each bone). To minimize the presented problem complexity, we decided to only predict 20 base bones including the hip bone, excluding finger bones completely. This results in an actual sum of 87 float values for each frame.

Using the method described above, we have recorded approximately 4.2 million individual OptiTrack frames. Thus the collected data for training the neural network summarizes to a size of about 20 GB. Figure 3 shows a fragment of the recorded data.



	0	1	2	3	4	5	6
0	Frame	Timestamp	TaskId	ProbandId	PS_SS_2019_Hip_POSITION_X	PS_SS_2019_Hip_POSITION_Y	PS_SS_2019_Hip_POSITION_Z
1	398780	1563538222.2692175	01	01	-0.09854848682880402	0.8931387066841125	-0.02167219296097754
2	398781	1563538222.2736356	01	01	-0.09846188127994537	0.8931452035903931	-0.0217666259765625
3	398782	1563538222.277632	01	01	-0.09838151931762695	0.8931528925895691	-0.021880494306612
4	398783	1563538222.281597	01	01	-0.09830460697412491	0.8931491374969482	-0.02198389172554016

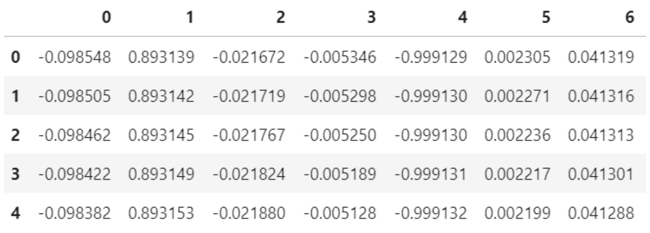
Figure 3: Extract from the collected data. Shows frame, timestamp, task ID, respondent ID, and X,Y,Z coordinates of hip position.

5 DATA PREPROCESSING

Before the data set could be used for training the neural network, the data had to be preprocessed.The first step was to ensure that data lost during data recording was reconstructed, as the OptiTrack stream is only available through UDP and hence does not implement any kind of flow control. Using linear interpolation it was possible to reproduce a completely missing frame and thus solving the above stated problem, even if this was only necessary once in the entire data set.

Secondly, we noticed that in some cases no values for certain position or rotation data were transmitted by OptiTrack. The problem of missing values was solved by linear interpolation, as well. Interpolation between the corresponding values of the predecessor and successor frames was performed to estimate the missing value.

Finally, linear interpolation was again used to concatenate the data subsets of the different test persons, resulting in a large data set containing all frames of all test persons. Additionally we removed all data rows not needed by the neural net. The resulting data set was slightly slimmer and contained only rotation and position information of the bones. The assignment of the raw data to the corresponding bones or joints is done exclusively in our Intercepter Client, which is presented in section. Figure 4 shows a fragment of the cleaned, finalized data.



	0	1	2	3	4	5	6
0	-0.098548	0.893139	-0.021672	-0.005346	-0.999129	0.002305	0.041319
1	-0.098505	0.893142	-0.021719	-0.005298	-0.999130	0.002271	0.041316
2	-0.098462	0.893145	-0.021767	-0.005250	-0.999130	0.002236	0.041313
3	-0.098422	0.893149	-0.021824	-0.005189	-0.999131	0.002217	0.041301
4	-0.098382	0.893153	-0.021880	-0.005128	-0.999132	0.002199	0.041288

Figure 4: Cleaned extract from the preprocessed data. Shows frame, timestamp, task ID, respondent ID, and X,Y,Z coordinates of hip position.

6 MOVEMENT PREDICTION SYSTEM

With the system latency value at hand, provided by our Latency Test Framework introduced in section 3, the second technical subsystem for predicting and rendering the actual body movements was built. Utilizing the presented data set in the previous section a neuronal network was designed and developed. For the development we used Google Brain's Tensorflow[34], which allows not only the simple and fast creation of neural networks but also the use of the high-level application programming interface (API) Keras[2] to create highly complex networks[29].

Neural Network

The developed network may be classified as a classical Fully Connected Neural Net(FQNN), also known as Deep Neural Net (DNN)[33]. DNNs are usually used for classification problems of all kinds[9, 32], although the problem presented in this paper cannot be considered one of these we decided to use a customized DNN.Another crucial factor in choosing the network architecture was the time required for interference. Since the presented problem is exceedingly time-critical, it was necessary to minimize the interference time - this matter will be further addressed in the next subsection. To achieve this we tried to make the network as simple as possible. In contrast to the classical and conventional implementation of DNNs, the network in this paper does not use the SoftMax function. Since a categorical representation of the probability distribution of the output yields no benefit. The network consists of 87 input neurons, which pass data to the first of two hidden layers. This hidden layer contains 4096 units and is fully connected to the second hidden layer of 8096 units. A dropout function has been added to the second hidden layer[10, 30] to battle *overfitting*. The dropout rate was deliberately chosen to be relatively low, at 20 percent, to avoid slipping in the under-learning range. Subsequently follows the output layer with built-in ReLu activation function[7]. Figure 5 depicts the essential structure of the developed neural network.

The aim of the network is to use the available 87 data points from the OptiTrack stream, to calculate 87 data points that are in the future displaced by a certain time. These 87 data points correspond to the position data of the hip and the rotation data of the other 21 joints in the frame. This procedure can be illustrated by using figure 4. The first line of the figure reflects the input (X) into the neural network. The last line shows the expected output (Y), in case the network should predict 4 frames into the future. Since the OptiTrack stream is available in 240Hz and the presented data was also recorded with this frequency, a prediction of 4 frames would correspond to a prediction of 12 milliseconds. For stochastic optimization we used a variant of the ADAM[15] Optimizer, which is included by Tensorflow. The training process was initiated with a learning rate of 1-e3 and a batch size of 512 samples. By using Keras callback functions[3], the learning rate could be dynamically adjusted during the training process. Depending on the validation accuracy, the learning rate was either increased or decreased up to a fixed threshold value. This process is comparable to classical learning rate decay in stochastic gradient descent method implemented in Tensorflow [35], but for our problem our implementation proved to be much more effective in dealing with optimization plateaus. The Mean Squared Error (MSE), as shown in formula 2, has successfully proven itself for the loss functions in our simulation environment.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2. \tag{2}$$

First experiments demonstrated that the developed network was already able to solve the given problem extraordinarily well. However, the Accuracy did not diverge clearly against a value, this was caused by the implemented dropout function in the last hidden layer. To fix this problem, we implemented another Keras callback function, which allowed the model to cancel the training process if performance deteriorated. In such a case, the previous version of the model was loaded and saved. This procedure is called Early Stopping (ES). Figure 6 shows the validation accuracy over 50 epochs without the implementation of the ES function. Using the above presented neural network we built various models, which differ in prediction value. The first model was built to predict 12 Frames into the future, which equals a time value of 48ms. This also equals the mean of the measured latency in section 3. Utilizing this model we are able to create a zero latency OptiTrack motion capturing environment. Besides the 12 frame model, we trained and optimized a 25 frame model, as well. 25 frames, again, equal to a time value of 100ms. With this model, we are not only able to compensate for system latency, but also predict actual future movements of human avatars. Both models are extremely well suited to their task, with the 12 frame model achieving a prediction accuracy of 94.2 percent on an unknown validation data set.The 25 frame model, on the other hand, still achieved a very good accuracy of 87.4 percent. To be able to test and study the developed models it was necessary to interrupt the OptiTrack stream, intercept the streamed frames, feed the model with the frames and return the prediction to the stream. To achieve this, we have developed our own Intercepter client, which is presented in the following subsection.

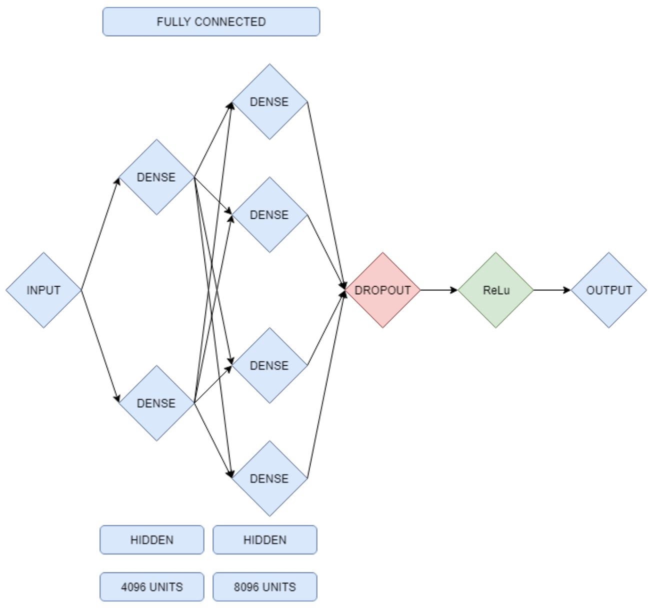


Figure 5: Shows a schematic representation of the developed network. The two hidden layers are fully connected and are composed of 4096 and 8096 units respectively. In combination with input and output layers, this amounts to a cumulative number of about 9 million trainable parameters.

Intercepter Client

The developed client has several time-critical tasks:

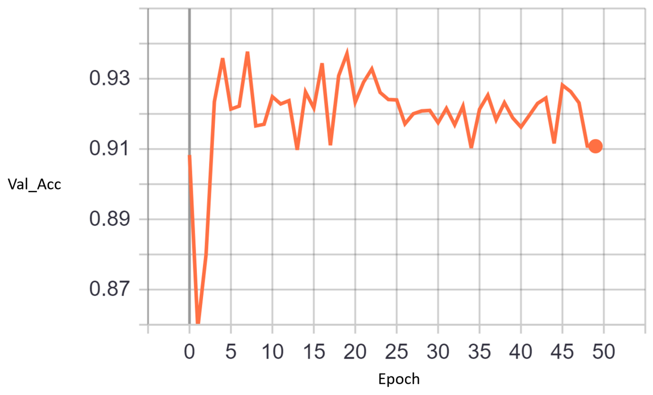


Figure 6: Showing validation accuracy of the trained network before implementing ES-function. The figure shows that a fixed number of epochs does not result in the optimized outcome.

(1) Unpacking the bit-stream provided by OptiTrack
(2) Feeding the raw data into the loaded model
(3) Waiting for model interference, accepting the prediction
(4) Repacking the prediction of the model into the bit-stream

This needed to be done for every single received packet, since one packet contains exactly one frame.This creates the time criticality already mentioned above. OptiTrack streams with a frequency of 240Hz, which means that every 4ms a new packet arrives at the Intercepter Client. The client has a maximum of 4ms seconds before a new packet has to be processed. During this time the client must have processed the packet completely and sent it on, otherwise the socket is blocked and the incoming packet can only be discarded. The discarded packages would cause the target application, in our case the Unity application, to tremble in the avatar representation. Due to the missing information, a continuous representation of the avatar would not be possible. Unpacking (1) and repacking (4) the data to a bit-stream takes the final client no significant amount of time, and thus can be neglected. Following the client has less then 4ms time to get the prediction from the model. To be able to to achieve this, the client pre-loads all used models (12 frame model and 25 frame model), thus having them ready at run-time. Additionally the clients sets the loaded model as the default Tensorflow graph, meaning the graph does not have to be rebuilt every time a prediction is needed. In addition to these measures, we evaluated the prediction time of various network architectures in advance. We evaluated the classical DNN structure used here, a classical RNN, an RNN based on CudnnLSTM units and an RNN based on CudnnGRU units. Figure 7 shows the evaluation of the measurements. It is shown that only a classical DNN structure can be considered for the presented problem, since all other architectures do not reliably remain below the 4ms limit.

7 EVALUATION STUDY

To evaluate the presented system we conducted an additional study. We invited 12 test persons (6f, 6m) with an average age of 24,5 (SD 3,0) to participate in the study. Requirements for the participants

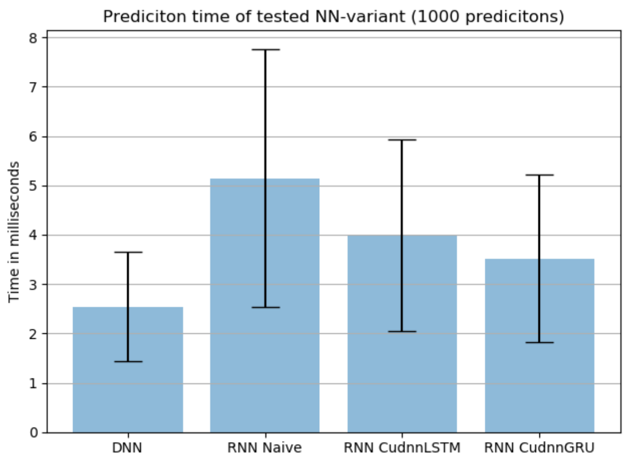


Figure 7: Shows the average time needed for different models to predict one single frame. Measurements rounded to one decimal. Each model needed to predict 1000 frames (n=1000) with 87 float values as input (X) and 87 float values as output (Y). Not considered in this graph is the time needed to train the corresponding model or prediction quality.

were identical to those presented in the data acquisition study. At the start of the evaluation study participants had to sign a consent form again to accept the collection of personal data as part of the used questionnaire and motion capturing.

The study consisted of 2 tasks with 3 conditions each. The three different tested conditions were:

(1) VR MoCap with 0ms prediction, real system latency
(2) VR MoCap with 48ms prediction, zero system latency
(3) VR MoCap with 100ms prediction, 52ms future prediction

Both tasks were performed by the volunteer in VR. For the study, a replica of the real room in which the subjects were at the time of the study was used. The first task was based on the task from the data collection study. In doing so, the test persons again had to simply perform movements from everyday life in VR, such as waving their hands. The second task consisted of a performance task in which the test persons were to complete a given task as quickly as possible. The given task consisted of the respondent touching alternately placed objects only using the same hand in VR. These objects were also present in the real world, this way the test person could experience real haptic feedback when touching the objects in VR. The count of contacts was recorded by the experimenter. The combination of conditions and tasks resulted in 6 distinct test scenarios for each respondent:

(1) T0: Interaction Task with 0 ms prediction
(2) T12:Interaction Task with 48 ms prediction
(3) T25:Interaction Task with 100 ms prediction
(4) P0: Performance Task with 0 ms prediction
(5) P12:Performance Task with 48 ms prediction
(6) P25:Performance Task with 100 ms prediction

After each scenario, the subjects had to complete a questionnaire and answer three short qualitative questions. The questionnaires were displayed directly in VR and the participants were able to complete them without having to remove the HMD. This procedure saves time and the respondent is not as easily influenced or distracted by the real world [27]. The questionnaire itself contained 3 questions of the IPQ[12] which have shown to have the highest reliability compared to similar standardized presence questionnaires[27]. Additionally we used 7 questions to evaluated the virtual limb ownership of the test subject. The used questions were originally designed to evaluated limb ownership of the hand in VR[28]. We adapted the questions to evaluated the limb ownership of the whole body. For each statement, participants chose a rating on a 5-point Likert scale ranging from 1 for *I strongly disagree* to 5 for *I strongly agree*. Table 1 shows all questions asked via questionnaire, as well as the means of the answers. The qualitative questions after each scenario were aimed to reveal model preferences and or remarks to the overall experiences. The following questions were verbally asked and recorded:

(1) QL1: Did you have the feeling that the virtual environment was in tune with reality?
(2) QL2: Did you have the feeling that your real movements matched your virtual movements?
(3) QL3: Did you feel nauseous during the last part of the study?

Upon finishing a task (T or P) we asked the participants which model they liked best to conclude the task. Once all the tasks had been completed, we asked them which model they liked most overall. The participants did not know which model they had tested, but named a unique key number that could be assigned to the model by the experimenter. Findings of the conducted study will be pointed out in the next section. Subsequently we discuss the finding in the section after next.

8 RESULTS AND FINDINGS

8.1 Data Analysis

In a first step, the collected data were examined for significance. Firtly, we evaluate the IPQ questionnaire(IPQ1,IPQ6 and IPQ10). The whole questionnaire can be considered as one entity and examined as such. The questions of the Performance (P) task and the Interaction task (T) were considered separately. Neither the distribution of the answers in the Performance task nor the distribution of the answers in the Interaction task could be demonstrated to be significant with the help of the Friedman test (p-value T-Tasks: > 0.9, p-value P-Tasks: >0.9). Figure 8 8 shows the calculated means as well as the corresponding standard derivation. Secondly, we examined the collected data for the limb-ownership (VL1 to VL7). Due to the structure of the collected data, it is not possible to evaluate the questionnaire as an single entity, it is necessary to examine each individual item. Before being able to analyze the data, the data set containing the VL-questions needed to be rank transformed. We used the ART method[40] available in R through the CRAN content delivery network[5]. After transformation, the individual measurements for each questions were entered into a repeated measure analysis of variance (ANOVA). Again, we separated the questions by the two performed tasks Interaction (T) and Performance (P). We found that VL4 in the Performance task *It seemed to me that* *my body was placed in the virtual world* significantly changed in dependencies of the used model,*p = 0.029*. Furthermore, question VL3 *My movements in virtual reality were performed by me* asked in the Interaction task has also revealed significant differences, *p = 0.03*. All other question did not reveal any significant differences. The above findings will be discussed in the section *Discussion and Conclusion*.

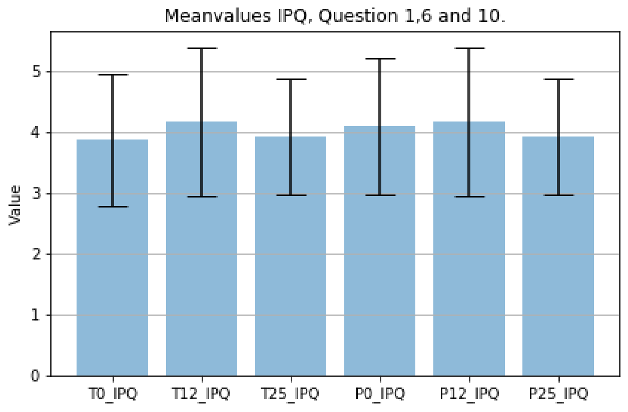


Figure 8: Shows IPQ means for each condition as well as SD as error bar. No significant difference between the variances could be proven.

Thirdly and lastly we analyzed the actual performance in the Performance task over all models.Since the performance task was a repeated measure, it has to be expected that a certain learning effect can be observed in the test persons during the measurement. To account for this effect, the sequence was considered an influencing factor in the data analysis. 9 shows the mean counts for each model, as well as the confidence interval for each bar. Although a certain trend could be recognized, ANOVA showed that there is no significant effect, *p = 0.53*.

8.2 Qualitative Analysis

All participants felt physically and mentally good after the study. The qualitative data indicate that none of the subjects felt nauseous or dizzy during or after the study. No *motion sickness* was induced by the use of either of our models (QLL3). Furthermore the qualitative feedback showed that the test persons had the feeling that the real world is depicted in the VR (QL2). Only one respondent answered the question in the negatively and went on to explain that especially the strange representation of the fingers had a strong negative influence on the immersion feeling. The problem described by the test person can probably be traced back to the very large tracking range of the OptiTrack system. At greater distances, the accuracy of the representation decreases due to transmission and differentiation problems. The motion capturing system is no longer able to clearly identify the individual fingers. All participants affirmed the question *Did you have the feeling that the virtual environment was in tune with reality?*(QL1). Particularly acclaimed was the realistic representation of the real world in VR. Surprisingly, when asked about the model that was most liked, half of the subjects preferred the 12F model for the Interaction task (T) and thus at least qualitatively confirmed the assumption that zero latency can increase the overall feeling of VR. Only three subjects preferred the 0F model for the Interaction task, another three subjects liked the 25F model best. There were no differences in the performance task(P), all three models were named by four participants each. Answering the concluding question as to which model the participants had liked best on the whole, six participants favored the 12F model again. Two participants chose the 25F model as the best model and another three participants chose the 0F model. One participant could not decide on a best model.

9 DISCUSSION AND CONCLUSION

Based on the evaluation of the IPQ it can be determined that the use of the presented models does not improve the immersion feeling. At the same time, however, it also shows that the above-mentioned immersion feeling is not worsened by the use of the models. There can be several possible explanations for this situation. One reason could be that ten out of twelve participants had no or very little previous experience in and with VR. It is possible that the participants were so impressed by the motion capturing system that they did not perceive the subtle differences in the representation of their own bodies. For the solution, respectively prevention, a stronger contrast between the conditions should have been chosen during the design of the conducted study. For the solution, respectively prevention, a stronger contrast between the conditions should have been chosen during the design of the conducted study. The differences between a 0 frame model and a 100 frame model, which corresponds to a prediction of 400ms, might have been noticed by a subject with very little previous experience. Another reason for the lack of significance may be the very small sample size (n=12). By increasing the sample size, effects could be shown which we are not able to prove significantly in the presented work.

The evaluation of the questions on limb ownership showed significant differences in the variance of the answers to questions VL4 (P) and VL3 (T). However, although a significant result was obtained by appropriate test procedures, there was no actually detected effect. Most likely the significance shown is attributable to a Type 1 error. Type 1 errors occur when randomly increased measurements lead to a rejection of the null hypothesis, although it should have been accepted. The assumption that there is a Type 1 error is further reinforced by the relatively weak significance at p-values around 0.03. A reduction of the accepted significance level, for example to 0.01, would reduce Type 1 error probability, but at the same time increase the error probability for a Type 2 error.

The evaluation of the Performance task showed no significant results. Although a general trend can be seen in the data and in figure 9, no universal effect may be established at this point. Three points could play a crucial role here. Firstly, the sample size is again too small, and the task could benefit enormously from an increase. Secondly, the performance task was not optimally planned by us. Instead of considering learning effects in the evaluation, a learning phase should have been implemented in the study. In this learning phase, the subjects could have familiarized themselves with the task and the environment. This phase would not be considered in subsequent evaluations. In this way, the actual performance of the test persons could be examined strongly decoupled from learning effects. Furthermore, test persons should be selected based on VR experience to prevent participants feeling intimidated or too impressed by the system. The last point influencing the outcome of this task is the chosen measurement. We only measure on variable - the times the test persons touches the objects. In order to be more reliable we could have established a second or third variable.

To conclude this paper, we shortly summarize: We successfully developed and implemented a reliable framework to test the latency of any motion capturing system coupled to a Unit application and a HMD. We successfully developed a neural network which is able to dependable predict human avatar movement in VR up to a value of 100ms. We successfully intercepted the available OptiTrack stream and feed the stream our own predicted values. Unfortunately, we were not able to evidently prove that a zero latency system is potentially able to increase the feeling of immersion, yet. In future work we will keep on working on the presented system. Firstly we need to develop a more suitable task scenario for the Performance task measuring more than only one variable. Secondly we want to improve the neural network and implement additional prediction values. Thirdly a proper study needs to be designed as already stated our chosen condition were probably too similar to be noticed by the participants. In a future design we will implement a condition which clearly distinguish itself from the 0 Frame baseline model.

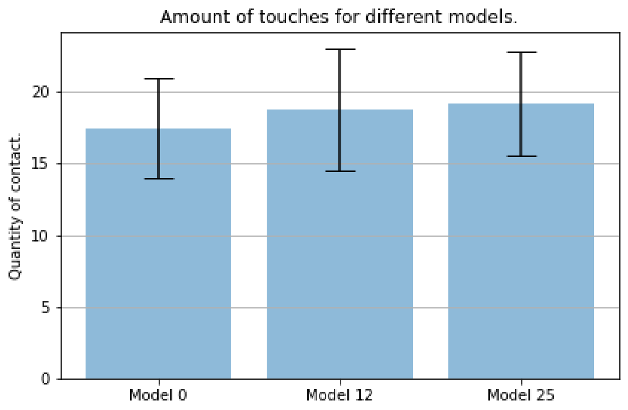


Figure 9: Shows the average hit count for the Performance Task (P), as well as SD as error. Although a certain trend is presumably, there is no actual significant effect provable.

REFERENCES

[1] Arduino. 2015. Arduino. https://www.arduino.cc/. Accessed on 2019-05-21.

[2] Francois Chollet. 2019. Keras. https://keras.io. Accessed on 2019-08-12.

[3] Francois Chollet. 2019. Keras. https://keras.io/callbacks/. Accessed on 2019-08-12.

[4] Philo Tan Chua, Rebecca Crivella, Bo Daly, Ning Hu, Russ Schaaf, David Ventura, Todd Camill, Jessica Hodgins, and Randy Pausch. 2003. Training for physical tasks in virtual environments: Tai Chi. In *IEEE Virtual Reality, 2003. Proceedings.* IEEE, 87–94.

[5] CRAN. 2015. The CRAN Projekt. https://cran.r-project.org/web/packages/ART/index.html. Accessed on 2019-08-30.

[6] Partha Ghosh, Jie Song, Emre Aksan, and Otmar Hilliges. 2017. Learning human motion models for long-term predictions. In *2017 International Conference on 3D Vision (3DV).* IEEE, 458–466.

[7] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics.* 315–323.

[8] Mar Gonzalez-Franco and Jaron Lanier. 2017. Model of illusions and virtual reality. *Frontiers in psychology* 8 (2017), 1125.

[9] Kun Han, Dong Yu, and Ivan Tashev. 2014. Speech emotion recognition using deep neural network and extreme learning machine. In *Fifteenth annual conference of the international speech communication association.*

[10] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580* (2012).

[11] HTC. 2015. VIVE Pro. https://www.vive.com/de/product/vive-pro/. Accessed on 2019-05-21.

[12] igroup. [n. d.]. igroup presence questionnaire. http://www.igroup.org/pq/ipq/index.php. Accessed on 2019-05-22.

[13] Xiaomo Jiang and Hojjat Adeli. 2005. Dynamic wavelet neural network model for traffic flow forecasting. *Journal of transportation engineering* 131, 10 (2005), 771–779.

[14] Rudolph Emil Kalman. 1960. A new approach to linear filtering and prediction problems. *Journal of basic Engineering* 82, 1 (1960), 35–45.

[15] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[16] Merel Krijn, Paul MG Emmelkamp, Roeline Biemond, Claudius de Wilde de Ligny, Martijn J Schuemie, and Charles APG van der Mast. 2004. Treatment of acrophobia in virtual reality: The role of immersion and presence. *Behaviour research and therapy* 42, 2 (2004), 229–239.

[17] Jingjing Liu, Shaoting Zhang, Shu Wang, and Dimitris N Metaxas. 2016. Multispectral deep neural networks for pedestrian detection. *arXiv preprint arXiv:1611.02644* (2016).

[18] Zhicheng Liu and Jeffrey Heer. 2014. The effects of interactive latency on exploratory visual analysis. *IEEE transactions on visualization and computer graphics* 20, 12 (2014), 2122–2131.

[19] Kin C Luk, James E Ball, and Ashish Sharma. 2001. An application of artificial neural networks for rainfall forecasting. *Mathematical and Computer modelling* 33, 6-7 (2001), 683–693.

[20] Julieta Martinez, Michael J Black, and Javier Romero. 2017. On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 2891–2900.

[21] Michael Meehan, Sharif Razzaque, Mary C Whitton, and Frederick P Brooks. 2003. Effect of latency on presence in stressful virtual environments. In *IEEE Virtual Reality, 2003. Proceedings.* IEEE, 141–148.

[22] Naturalpoint. 2015. Optitrack - Motion Capturing System. https://optitrack.com/. Accessed on 2019-05-21.

[23] Matthew Price and Page Anderson. 2007. The role of presence in virtual reality exposure therapy. *Journal of anxiety disorders* 21, 5 (2007), 742–751.

[24] Simon Riches, Soha Elghany, Philippa Garety, Mar Rus-Calafell, and Lucia Valmaggia. 2019. Factors Affecting Sense of Presence in a Virtual Reality Social Environment: A Qualitative Study. *Cyberpsychology, Behavior, and Social Networking* (2019).

[25] Matthew NO Sadiku. 2014. *Elements of electromagnetics.* Oxford university press.

[26] Maria V Sanchez-Vives and Mel Slater. 2005. From presence to consciousness through virtual reality. *Nature Reviews Neuroscience* 6, 4 (2005), 332.

[27] Valentin Schwind, Pascal Knierim, Nico Haas, and Niels Henze. 2019. Using Presence Questionnaires in Virtual Reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems.* ACM, 360.

[28] Valentin Schwind, Lorraine Lin, Massimiliano Di Luca, Sophie Jörg, and James Hillis. 2018. Touch with foreign hands: the effect of virtual hand appearance on visual-haptic integration. In *Proceedings of the 15th ACM Symposium on Applied Perception.* ACM, 9.

[29] Rajalingappaa Shanmugamani. 2018. *Deep Learning for Computer Vision: Expert techniques to train advanced neural networks using TensorFlow and Keras.* Packt Publishing Ltd.

[30] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from

overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.
[31] Mate Szarvas, Akira Yoshizawa, Munetaka Yamamoto, and Jun Ogata. 2005. Pedestrian detection with convolutional neural networks. In *Intelligent vehicles symposium*. 224–229.
[32] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. 2013. Deep neural networks for object detection. In *Advances in neural information processing systems*. 2553–2561.
[33] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).
[34] Google Brain Team. 2015. TensorFlow. https://www.tensorflow.org/. Accessed on 2019-05-21.
[35] Google Brain Team. 2015. TensorFlow. https://www.tensorflow.org/api_docs/python/tf/keras/optimizers/SGD. Accessed on 2019-05-21.
[36] Unity Technologies. 2005. Unity. https://unity.com/. Accessed on 2019-05-21.
[37] Denis Tomè, Federico Monti, Luca Baroffio, Luca Bondi, Marco Tagliasacchi, and Stefano Tubaro. 2016. Deep convolutional neural networks for pedestrian detection. *Signal processing: image communication* 47 (2016), 482–489.
[38] Iis P Tussyadiah, Dan Wang, Timothy H Jung, and M Claudia tom Dieck. 2018. Virtual reality, presence, and attitude change: Empirical evidence from tourism. *Tourism Management* 66 (2018), 140–154.
[39] Hans Georg Unger and Johann Heyen Hinken. 1991. *Elektromagnetische Wellen auf Leitungen*. Hüthig.
[40] Jacob O Wobbrock, Leah Findlater, Darren Gergle, and James J Higgins. 2011. The aligned rank transform for nonparametric factorial analyses using only anova procedures. In *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 143–146.
[41] Jiann-Rong Wu and Ming Ouhyoung. 2000. On latency compensation and its effects on head-motion trajectories in virtual environments. *The visual computer* 16, 2 (2000), 79–90.

10 APPENDIX

Question	Questionnaire	T0	T12	T25	P0	P12	P25
In the computer generated world I had a sense of being there.	IPQ1	3.8±0.7	4.1±1.2	3.9±1.0	4.1±1.1	4.0±1.1	4.25±0.8
I feel present in the virtual space.	IPQ6	3.6±1.6	4.0±1.6	3.8±0.9	3.8±0.9	3.8±1.0	3.76±1.3
I was completely captivated by the virtual world.	IPQ10	4.1±1.1	3.8±1.5	4.0±0.9	3.6±1.8	4.0±1.5	3.9±1.0
I had the feeling that my virtual body was in the same position as in reality.	VL1	3.1±1.3	3.6±1.2	4.1±1.6	3.8±1.3	4.0±1.4	3.6±1.3
I had the feeling that the touch in reality was created by the touch in the virtual world.	VL2	4.6±0.9	4.5±0.7	2.6±1.6	4.3±0.9	3.8±1.5	4.3±0.9
My movements in virtual reality were performed by me.	VL3	3.9±0.8	4.3±0.7	4.0±0.9	3.9±0.5	5.1±0.9	4.3±0.8
It seemed to me that my body was placed in the virtual world.	VL4	3.2±1.2	2.7±1.4	2.8±1.0	3.1±1.3	3.1±0.9	3.4±1.3
The virtual body resembled my real body in form and appearance.	VL5	3.2±1.3	3.3±1.3	3.4±1.0	3.3±1.1	3.4±1.0	3.1±1.5
My feelings were influenced by what I saw in virtual reality.	VL6	3.3±1.7	3.9±0.9	3.2±1.2	3.8±1.2	3.8±1.2	3.8±1.1
My virtual body felt like my real body.	VL7	4.1±0.8	3.8±1.5	3.8±0.9	4.1±1.1	3.9±1.0	4.3±0.6

Table 1: Questions asked after each test case. Also showing the means of the answers, as well as SD. Test persons answered these question directly in VR.