Virtual Reality (2024) 28:65
https://doi.org/10.1007/s10055-024-00962-9

ORIGINAL ARTICLE

Long short-term memory prediction of user's locomotion in virtual reality

Jesus Mayor[1] · Pablo Calleja[1] · Felix Fuentes-Hurtado[1]

Received: 20 July 2023 / Accepted: 25 October 2023 / Published online: 5 March 2024

Abstract
Nowadays, there is still a challenge in virtual reality to obtain an accurate displacement prediction of the user. This could be a future key element to apply in the so-called redirected walking methods. Meanwhile, deep learning provides us with new tools to reach greater achievements in this type of prediction. Specifically, long short-term memory recurrent neural networks obtained promising results recently. This gives us clues to continue researching in this line to predict virtual reality user's displacement. This manuscript focuses on the collection of positional data and a subsequent new way to train a deep learning model to obtain more accurate predictions. The data were collected with 44 participants and it has been analyzed with different existing prediction algorithms. The best results were obtained with a new idea, the use of rotation quaternions and the three dimensions to train the previously existing models. The authors strongly believe that there is still much room for improvement in this research area by means of the usage of new deep learning models.

Keywords Virtual reality · User prediction · Locomotion · Deep learning

1 Introduction

Virtual reality (VR) technologies are widely used, and it is a choice for many companies to develop virtual experiences or video games. However, speaking in terms of locomotion, when a user walks naturally, it is only a matter of time before the user collides with the boundaries. This major constraint forces companies to look for solutions and adapt their designs to the limitations of the physical workspace.

Through human ingenuity, alternatives have been developed to avoid these limitations. Some examples of these new locomotion methods could include the use of controllers or the inclination of the body (Zielasko et al. 2016; Langbehn et al. 2018). However, the choice between locomotion methods is not trivial, as it can directly affect user perception by varying factors such as presence, cybersickness, or usability (Mayor et al. 2019).

One of the most promising solutions to workspace limitations are the so-called Redirected Walking (RDW) locomotion methods (Razzaque et al. 2001). These methods are intended to preserve all the benefits of natural locomotion in physical space while avoiding its limitations. Taking advantage of the real movement of the user, RDW methods attempt to trick the human brain by applying subtle modifications to the virtual world. This effect is achieved through intelligently designed gain algorithms prepared to avoid the limits of the workspace (Suma et al. 2012).

In turn, different strategies are proposed to avoid the limits. For example, it is called *steer to center* if the redirection is to the center of the workspace, *steer to orbit* if the motion of redirection adheres to a specific orbit (Razzaque 2005), and *steer to multiple targets* if the redirection is to multiple central points (Hodgson and Bachmann 2013). This idea has gone so far as to use deep learning (DL) for the selection of the most optimal redirection target (Lee et al. 2019). There are even solutions using reinforcement learning to avoid multiple simultaneous users meeting each other in the same workspace (Lee et al. 2020).

Jesus Mayor, Pablo Calleja and Felix Fuentes-Hurtado have contributed equally to this work.

✉ Jesus Mayor
jesus.mayor@upm.es

Pablo Calleja
p.calleja@upm.es

Felix Fuentes-Hurtado
felix.fuentes@upm.es

[1] Sistemas Informáticos, Universidad Politécnica de Madrid, Alan Turing, 28031 Madrid, Spain

To make it all work, the gain algorithms should follow different and well-defined strategies. However, a RDW method can be composed of a single algorithm (e.g., curvature gain Bölling et al. 2019) or multiple algorithms (Nescher et al. 2014; Grechkin et al. 2016). Four basic strategies can be identified in the gain algorithms (apart from other more heterogeneous variants. e.g. strafing gain You et al. 2022): curvature gain, translation gain, rotation gain, or deviation gain (Mayor et al. 2021). However, it is clear that there is another essential side. The third pillar to make gain algorithms work is an accurate prediction of where the user is heading.

Currently, VR technologies have started to use machine learning methods with different proposals (Lee et al. 2019; Brenneis et al. 2021). To predict where the user is heading, there exists some research that has yielded great results, but more research is needed to strengthen this claim (Bremer et al. 2021).

In particular, Recurrent Neural Networks (RNN) are used to predict future user positions given a myriad of input variables (Dupond 2019). RNNs are a type of neural network designed to process sequential data by leveraging recurrent connections. Unlike feedforward neural networks, RNNs possess feedback connections that allow them to capture temporal dependencies within a sequence. They operate by maintaining hidden states that retain information from previous time steps, thereby enabling the network to learn and reason about long-term dependencies. This recurrent nature endows RNNs with the ability to model sequential data with variable-length input and output sequences.

However, RNNs have some weaknesses. One of them is the lack of long-term memory. To solve that, Long Short-Term Memory (LSTM) networks were proposed (Hochreiter and Schmidhuber 1997). LSTM networks are a specialized variant of RNNs that effectively addresses the vanishing gradient problem associated with training deep RNN architectures. LSTMs introduce gated mechanisms, including input, forget, and output gates, which enable the networks to selectively remember or forget information at different time steps. These gates regulate the flow of information, allowing LSTMs to capture long-term dependencies while mitigating the vanishing or exploding gradient issues that hinder traditional RNNs. The memory cell within an LSTM unit acts as a key component that effectively stores and retrieves relevant contextual information, making it well-suited for modeling complex temporal dependencies.

The sequential nature of trajectory data in VR environments naturally lends itself to the application of RNNs and LSTMs. By processing historical trajectory data, RNNs and LSTMs can learn to model the inherent dynamics and dependencies present in the data. This learned representation can be used to predict the future trajectory of objects in the VR environment. The ability of RNNs and LSTMs to capture long-term dependencies and handle variable-length sequences makes them well-equipped to meet the challenges associated with user trajectory prediction.

This manuscript aims to improve previous results in user movement prediction by presenting an adapted prediction model using neural networks. It aims to achieve this with a minimal hardware requirement, close to the commercial virtual experiences. To do so, we implemented a newly created dataset with three virtual experiences. They have been designed with different proposes to record movement. Thus, the datasets are generated through experiments with real humans using these experiences. In each experience, locomotion is designed to be applied in a different way by mimicking different common VR situations.

In this way, this manuscript raises two questions:

- **RQ1**: Are the new datasets suitable to be evaluated with previous locomotion prediction algorithms?
- **RQ2**: Will the adapted prediction model predict better than previous locomotion algorithms?

To answer these questions, we implemented some common algorithmic solutions used to predict user locomotion in VR. Furthermore, we also implemented the deep learning algorithm presented by Bremer et al. (2021) which will be detailed in the next section.

2 State of the art

In the VR sector, we can find that user predictions have gained momentum in recent years. An example of this could be the interaction with the gaze. Some works try to predict the gaze of the user to achieve interactions where the gaze is looking at (Hu 2020). On the other hand, other papers are trying to predict where the user will look, concluding that dynamic elements are interesting elements in the scene to make gaze predictions (David-John et al. 2021).

The prediction of the user's movements in other areas also follows the same line. For example, Corona et al. (2020) proposed a context-aware motion prediction architecture that incorporates interactions with objects and humans. With a motion capture suit, they use a semantic graph model with a graph attention layer that is fed into an RNN to predict future human movements. Breuer et al. (2019) used a combination of LSTMs and CNNs to predict the future trajectories of traffic participants in autonomous driving systems.

In the context of VR, an example is the work of Gamage et al. (2021). Who used regression models to predict the path of the hands in VR and real-time. On the side of deep learning, Strauss et al. (2020) uses reinforcement learning to dictate the gain algorithms of redirection when using RDW methods. Finally, Cho et al. (2018) implement an

LSTM network to predict a user's future path based on their past position and facing direction data for efficient redirected walking in limited-sized rooms.

As mentioned above, RDW methods are highly supported by accurate user movement prediction. This prediction method could be classified as scripted, predictive, or reactive (Hirt et al. 2022a, b).

A scripted prediction involves a fixed path for the user to follow, thus simplifying the prediction made to a selection of a pre-defined set of paths (Zmuda et al. 2013; Nescher et al. 2014). In some cases, the prediction is used only to help to choose between two different corridors, as in the experiment presented by Nescher and Kunz (2013). Such is the need for these algorithms to have a pre-defined path that there are authors who have developed automatic algorithms for path extraction from a walkable mesh generation (Zank and Kunz 2017). However, in most virtual experiences, it is not possible to configure or pre-define a set of interesting positions to make these prediction models work. Some initiatives begin to form predictions from the user's historical movement data in the scene, although again this does not serve as a general solution, as it requires prior user data to work (Fan et al. 2023). In addition, a user outside of an experimental environment can change direction spontaneously, breaking with the nature of this type of prediction.

On the other hand, it is called a predictive prediction if is based on an approach to the path of the user in real-time (without being previously pre-defined by the developer). However, predictive approaches are exotic and least investigated, as trajectory calculation is more complex and is also prone to errors due to user spontaneity (Zank and Kunz 2016, 2015). Predictive methods can be really useful to approximate the trajectory to another desired one, like in the above-mentioned *steer-to-orbit* method (Razzaque 2005).

However, the most common prediction is reactive. Reactive predictions operate without prior knowledge of the path or a future path estimate. Therefore, this type of prediction relies on previous information captured from the user's movement to predict future positions. Therefore, it is beginning to become common to create a separation of reactive prediction into two modalities: short-term and long-term (Bremer et al. 2021; Hirt et al. 2022a; Stein et al. 2022). Short-term predictions are those that estimate the next instant of a discretized time, meanwhile, long-term predictions are those that estimate the future position of the user in a pre-defined and larger time window. These may appear to be very similar cases, but different heuristics are commonly sought to solve both.

As the prediction algorithms are numerous, we will only mention in-depth those used for the experiment developed in this manuscript and relate them to other existing methods.

2.1 Non-deep learning algorithms

Algorithmic solutions to motion prediction have been widely designed in the field of virtual reality. Their use has been common as an internal mechanism to avoid head-mounted display (HMD) tracking errors. Several of these were compiled and compared in the study of van Rhijn et al. (2005). In this study, they analyzed the Extended Kalman filter, the Unscented Kalman filter, the Particle filter, and the Linear Time-Invariant filter. The study put a lot of emphasis on the latency produced by the algorithms, as they are applied at a low level together with the device tracking.

LaViola (2003) compared the Kalman filter and the extended Kalman filter with the Double-exponential smoothing algorithm (DES). His experiment showed that DES runs approximately 135 times faster than the other two predictors. The root-mean-square error in DES obtains approximately the same results.

This algorithm is not only used to smooth the HMD tracking system; it is also used at higher abstraction levels to predict user movement. For example, in the study conducted by Nescher and Kunz (2013), DES is used to select between two different paths that support scripted prediction methods. This method was also used in other experiments without predefine the path in a reactive way as part of a RDW method (Mayor et al. 2021). Thus, the DES prediction seems to be a really suitable method for being used in RDW methods. The DES method applied to a position delta vector $\vec{w}_t$ (between the previous frame and the current frame) and a prediction delta vector $\vec{s}_t$ is defined as:

$$\vec{s}_0 = \vec{w}_0 = 0$$
$$\vec{s}_t = \alpha\vec{w}_t + (1-\alpha)(\vec{s}_{t-1} + \vec{b}_{t-1})$$
$$\vec{b}_t = \beta(\vec{s}_t - \vec{s}_{t-1}) + (1-\beta)\vec{b}_{t-1}$$

This method is based on a double linear interpolation using $\vec{b}_t$ as an intermediate step for the second interpolation. Nescher and Kunz (2013) defines $\alpha \in \mathbb{R}(0,1)$ as the *smoothing factor* and $\beta \in \mathbb{R}(0,1)$ as the *trend smoothing factor*. As $\vec{s}_t$ is a predicted delta vector of position, to calculate the future predicted position $\vec{p}_{t+1}$, we should add it to the current position $\vec{p}_t$. Smaller values of α will make the prediction less affected by new input data, resulting in the permanence of the prediction trend. Higher values of α will make the prediction more reactive to new input data and less sensitive to the trend of the previous samples. The DES algorithm will be used as a comparison in this manuscript due to its promising results.

The above-mentioned linear interpolation could also be considered to be the simplest reactive prediction method used to extrapolate. In terms of latency or response speed, linear extrapolation (LE) is used as a basic algorithm in some experiments as a baseline (Bremer et al. 2021). It will also be used as the simplest algorithmic baseline in our experiment.

However, LE should not be a good candidate, as it may have a large numerical error compared to other more sophisticated solutions. For a short explanation, this algorithm only uses the last frame with respect to the current frame positions. It is based on the idea that the user continues to do the same movement without taking into account the trends of those movements. Taking into account the delta vector of position $\vec{w}_t$ and the predicted delta vector of position $\vec{s}_t$, it can be expressed as $\vec{s}_t = \vec{w}_t$. Another way to understand it is that it would be equivalent to using DES with the value of $\alpha = 1$, completely ignoring the trend.

2.2 Deep learning prediction algorithms

Currently, deep learning techniques have been explored and implemented in a lot of research areas outperforming the previously achieved results. As mentioned above, reactive predictions using neural networks are beginning to emerge. The results are promising, but very few research papers support this hypothesis. To the best of the authors' knowledge, there is only one DL model that has been used in two different articles to draw their results (Bremer et al. 2021; Stein et al. 2022). This article will compare the existing DL model and explain it in depth.

In the first paper by Bremer et al. (2021), the presented model includes eye tracking and a body-wearable sensor. In addition, this model had two different designed heuristics depending on whether it was prepared for short-term or long-term predictions. These heuristics are designed so that the initial coordinates of the 3D scene do not matter and the model can learn correctly. However, the approach of our experiment will be closer to commercial use, in which no additional sensors are present. Therefore, due to the complexity of the Bremers model, a simplified version will be explained here, including only HMD tracking information.

Both heuristics use a rotation matrix $\mathbf{R}(\theta)$ to transform the data as incoming features in a specific reference system.

$$\mathbf{R}(\theta) = \begin{vmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{vmatrix}$$

As features of both models (long and short), we have ($\vec{W}_{t+n} \in \mathbb{R}^2$) as the future position predicted, ($\vec{V}_t \in \mathbb{R}^2$) as the velocity using, ($\Psi_t \in \mathbb{R}$), and ($\Phi_t \in \mathbb{R}$) (uppercase letters mean that we are using the new reference system).

$$\vec{W}_{t+n} = \mathbf{R}(-\theta)\vec{w}_{t+n}$$
$$\vec{V}_t = \mathbf{R}(-\theta)\vec{v}_t$$
$$\Psi_t = \psi_t - \theta$$
$$\Phi_t = \phi_t - \theta$$

The long-term heuristic uses the arithmetic mean of the Yaw axis ($\theta_l = \overline{\psi}_{t-1} \in \mathbb{R}$) of the HMD in a specific time window of 2.5 s of ($n = 50 \in \mathbb{Z}$) samples. For the authors, this window contains 2.5 s of samples to predict the same extent. Long story short, the incoming features of the model accumulate a mean yaw of 2.5 s to predict the same 2.5 s in the future.

On the other hand, the short-term heuristic, instead of using the mean of the HMD yaw angle, works by using the angle of velocity ($\theta_s = \angle(\vec{v}_{t-2}, (\begin{smallmatrix}0\\1\end{smallmatrix})) \in \mathbb{R}$). As we can see, the angle of the velocity of two steps before was used to apply it as the new reference system.

Tanking into account those heuristics to aim for long or short-term predictions, the used LSTM architecture had the following characteristics:

- Depending on whether the model includes eye or body tracking, from 4 to 7 input features.
- Two layers of 64 hidden units each.
- A dropout layer with ($p = 0.3$).
- Dense linear output with two values, giving the prediction of $\vec{W}_{t+n}$
- It is trained with 20 epochs, a learning rate of 0.003, and 50 timesteps.

Instead of directly computing the results with the mean absolute error as a single value, the mean displacement error (MDE) is used. It is calculated between the label of $\vec{W}_{t+n}$ (with two variables or dimensions) and the predicted values (in previous algorithms we called them $\vec{s}_t$, also with two variables).

Their results show that the use of extra sensors is only significant when predicting with a long-term heuristic (Bremer et al. 2021). Furthermore, in subsequent experiments (Stein et al. 2022), they continued to explore this idea, concluding that the use of eye-tracking data achieves greater precision with a slight improvement with respect to the non-usage of this technology and their DL model.

Their latest publications focus on this line to continue comparing different eye-tracking devices (Stein 2021), but they highlight that different technologies could produce more or less latency. Therefore, this could modify the results of the pre-trained models depending on the technology used. As mentioned above, this manuscript aims to avoid the use of additional hardware in combination with

HMD achieving improvements in the predictions made. In this way, this hardware dependency will be avoided.

3 Methods

This article proposes to analyze different predictive algorithms using real user movement data. For this reason, the experimentation has been carried out in two phases:

- Dataset acquisition: Dataset generated through a realistic use of different virtual experiences focused on obtaining a wide variety of data in different environments. The data obtained should be used to compare and correctly evaluate different prediction algorithms.
- Data analysis: We will analyze the prediction algorithms explained in the state-of-the-art (see Sect. 2; LE, and DES). Furthermore, this analysis will include the LSTM (Bremer et al. 2021) method and our own adaptation.

3.1 Data acquisition

In order to characterize in a complete way the users when performing movements in virtual environments, three different virtual experiences have been proposed. These experiences have been developed with Unity 2021.1.3f1 and they try to follow hard-to-track scenarios with elements found in commercial VR experiences (all code could be downloaded on Github Mayor et al. 2023). The objective pursued behind each experience is defined below.

- Scene laboratory (SL). This scene depicts a warehouse with boxes on the floor. The scene lacks interactions or defined objectives. In this way, it is impossible to define points of interest to script the prediction. Test subjects were asked to search for 3 min for a non-existent gnome in the warehouse. The non-existent gnome emitted a delocalized sound, encouraging the subjects to walk randomly without a clear goal during this experimentation (see Fig. 1A).
- Shooter forest (SF). The user should walk looking for faraway red targets to shoot. In this example, there are clearly defined targets that could be tracked; however, the movement does not necessarily have to follow these targets. In this way, there is another situation where it is difficult to script the prediction (Fig. 1B).
- Escape room (ER). This is a clear example of a VR game. In fact, this is an adapted version of a free-to-use Unity demo “VR Beginner: The Escape Room” (because the original version allowed the user to be teleported, breaking our requirements). In our version, the user should walk to explore different elements to solve a puzzle. Although there is a logical sequence of actions that could lead the user to perform specific movements, exploration may not lead the user to follow specific movements (Fig. 1C).

3.1.1 Experimental setup for data acquisition

All experiments were performed in the same room, under the same conditions. The workspace within this room was $8m \times 8m$ due to physical limitations. The device used for all experiments was the Oculus Quest 2, which uses inside-out tracking and is also approximately limited to that space.

Since the three experiences were created with Unity, a system capable of recording the relevant data in a *.csv* file was developed. This system consisted of two scripts adapting an Observer design pattern in the Unity framework. One was placed on each relevant object in the scene, and the other was in charge of accumulating the data provided by each of the previous scripts. Everything worked thanks to the use of the Unity XR Plugin Management package with Oculus support and UnityEngine.XR library that allowed us to access the device data. All the data finally extracted are described in Sect. 3.1.3.

3.1.2 Participants and procedures

The experiment had 44 participants, the mean age was 21.91 years, with a standard deviation of 5.65 and a median age of 20. The sample includes 33 men and 11 women, mainly university students from the *Universidad Politécnica de Madrid*.

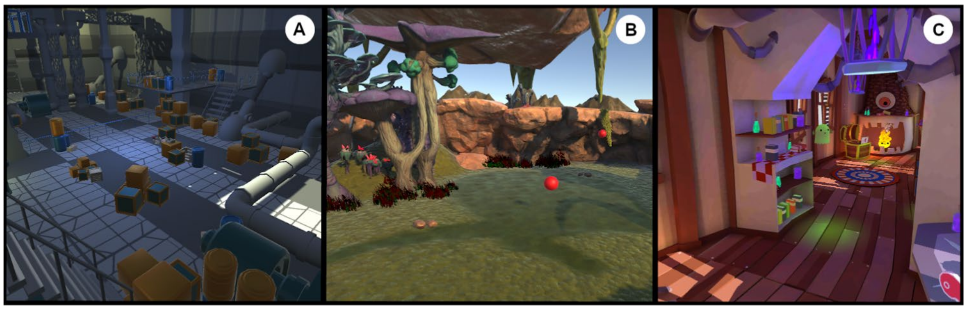

Fig. 1 General view of each VR experience used to capture the motion data

These participants were previously asked about their overall use of VR technology and their weekly dedication to video games in general. Table 1 shows the responses divided into time bands in relation to this question.

Among the participants, 37 stated that they did not have their own HMD (84.09%). Of those who did (remaining 15.91%), 4 had a Cardboard device and 3 had higher-end devices.

The experiment followed the same conditions for all subjects in terms of procedure, supervising person, and I/O devices used. A disinfecting procedure was also followed between experiments to ensure a COVID-free environment. In order to provide the same instructions to all users, they were previously written and read during experimentation.

To avoid that the order of virtual experiences may affect the results of the experiment, the experiment varied the order of the displayed experiences in a balanced way following the Latin square algorithm (Bradley 1958).

According to current legislation, all users gave their consent to the processing of their data in anonymized form for the execution of this article.

3.1.3 Dataset description

All scenes were recorded with a scale factor of 1.2 to fit the size of the room of the experiment. This recording took place approximately every 0.1 s, creating 10 records per second. With this procedure, the resulting dataset weighted around 266 MegaBytes in 275666 total samples. The dataset can be accessed on the Zenodo platform (Mayor et al. 2023).

Since it is recorded inside a game engine and all records take place inside their processing, the timestamp is written down for each register (Time_sice_startup field). Additionally, the anonymized identification of the user is recorded (User field). All recorded data about the device are directly captured from its information in that frame.

The dataset includes the following characteristics for Oculus Quest 2 HMD and each controller.

- DevicePosition (x, y, z): Position recorded (previously expressed as $\vec{p}_t \in \mathbb{R}^3$).
- DeviceRotation (w, x, y, z): Rotation expressed with a quaternion ($q_t \in \mathbb{H}$ used in our new model).
- Forward (x, y, z): The unit vector that points to the specific device in the forward direction ($\vec{f}_t \in \mathbb{R}^3$ used in our new model. It can also be obtained by rotating (0, 0, 1) with the quaternion q_t).
- DeviceVelocity (x, y, z): Linear velocity of that device in that frame. It represents the rate of change in position (equivalent to $\dot{p} \in \mathbb{R}^3$).
- DeviceAcceleration (x, y, z): Linear acceleration of that device in that frame (equivalent to $\ddot{p} \in \mathbb{R}^3$).
- DeviceAngularVelocity (x, y, z): The angular velocity vector in that frame of the device is measured in radians per second (if the rotation of each of the independent axes were Euler's rotation defined by $e \in \mathbb{R}^3$, this is equivalent to $\dot{e} \in \mathbb{R}^3$. Euler angles can be obtained from q_t).
- DeviceAngularAcceleration (x, y, z): The angular acceleration at that frame (equivalent to $\ddot{e} \in \mathbb{R}^3$).

To enrich the dataset in order to be analyzed in future research has also been defined into the scenes some interesting goals. In the virtual shooter experience, all targets were recorded as goals. In the virtual escape room experience, there were so many intractable elements, but only the key objects were defined as goals and recorded. However, in the scene laboratory, as does not have a real objective we could not define a goal. All of these goals left the following data in their respective recorded files.

- GoalName (x, y, z): Position of that goal. If the element is static, the same position will always be recorded.
- GoalName_Quat (w, x, y, z): As in the previously defined fields, a rotation is expressed as a quaternion.
- GoalName_LocalScale (x, y, z): Scale of that element locally related to its parent in the hierarchy. They have no relatives in their hierarchy, so it is the real scale.

3.2 Data analysis

The results achieved by Bremer et al. (2021) with their DL model seem to be really promising. However, there are specific issues that lead us to believe that this DL model could be improved. For example, the usage of Euler angles to fit the model, since it is not continuous data. Euler angles can jump between 0° and 360°, which means the same thing. This kind of data could be tricky to learn by artificial neurons to create effective learning.

Table 1 Sample subjects and their percentages of total VR use

Hours	0	0–2	2–5	5–10	+10
Total VR usage	17 (38.64%)	3 (6.82%)	15 (34.09%)	6 (13.64%)	3 (6.82%)
Weekly videogame	7 (15.91%)	19 (43.18%)	7 (15.91%)	4 (9.09%)	7 (15.91%)

Also, the weekly hours spent playing video games

In addition, they used an arithmetic mean of the yaw angle as the long-term reference system. This kind of mean using circular values does not work well, since the average of two values, 0° and 360° using the last example, the result should not be 180°. This could be solved using a circular mean, really common in the statistics field; in this way, the numerical error could be mitigated.

Thus, showing the limitations of the proposed Bremer's method, this section aims to improve and prepare the dataset for experimentation with the different prediction models.

3.2.1 Our new quaternion-based predictor

This manuscript tries to improve the model using a completely different approach without changing the input of the neural network architecture. The need to use Euler angles for Bremer et al. (2021) arises from the fact that they want to simplify the data sent to the neural network. If the dimensionality is reduced to the X-Z plane, with the Y axis (zenithal axis), the rotation angle yaw (ψ) is needed as a feature. Since the prediction is made on that plane, that sounds like a good solution.

Our approach takes the opposite direction. Since dimensional reduction to the X-Z plane brings certain problems, we wanted to train the DL model in three dimensions. In this way, although the features sent to the model are more complex, we do not deprive the neural network of some information to find the best solution. Moreover, inside computer graphics and many more areas, the rotations are operated using quaternions to avoid numerical errors.

The rotations expressed with quaternions, although more difficult for humans to understand because they belong to complex numbers, solve many of the problems introduced by the Euler angles. In this case, despite the fact that the quaternions express the same circular data as the Euler version, the values do not jump to express a really close rotation value (like the previous example of 0° and 360°).

Keeping this idea in mind, this manuscript tried to follow the same long-term heuristic but in three dimensions. In this case, the rotation expressed by the quaternion has been averaged using the previous samples to work as a new reference system for the input data and to predict the future. Since quaternions are not regular vectors, an average cannot be obtained by taking a weighted mean. To correctly average the quaternions, the Markley et al. (2007) algorithm has been used resulting in the average quaternion ($\overline{q}_{t-1} \in \mathbb{H}$) which is applied in a certain amount of previous ($n \in \mathbb{Z}$) samples prior to the current sample ($t \in \mathbb{Z}$).

Taking into account that ($D_t \in \mathbb{H}$) is the current rotation, ($\vec{F}_t \in \mathbb{R}^3$) is the forward vector, ($\vec{V}_t \in \mathbb{R}^3$) is the current velocity, and ($\vec{W}_t \in \mathbb{R}^3$) positional delta are features of our model under the new reference system. Homonymous variables in lowercase will be those without already following the reference system. In this way, $\vec{W}_t + n \in \mathbb{R}^3$ will be the labels, so they are three coordinates that will be predicted by the neural network, following the next equations.

$$\vec{W}_{t+n} = \overline{q}_{t-1}^{-1}\ \vec{w}_{t+n}\ \overline{q}_{t-1}$$
$$\vec{V}_t = \overline{q}_{t-1}^{-1}\ \vec{v}_t\ \overline{q}_{t-1}$$
$$\vec{F}_t = \overline{q}_{t-1}^{-1}\ \vec{f}_t\ \overline{q}_{t-1}$$
$$\vec{W}_t = \overline{q}_{t-1}^{-1}\ \vec{f}_t\ \overline{q}_{t-1}$$
$$D_t = q_t\ \overline{q}_{t-1}^{-1}$$

Since the prediction outcoming from the trained neural network ($\vec{S}_t \in \mathbb{R}^3$ referring to the current prediction of the future $\vec{W}_t + n \in \mathbb{R}^3$) is made in the new reference system, to evaluate this reference system, the prediction must be reverted by applying the opposite operation. To evaluate both models under the same metric, we have also used the MDE explained in Sect. 2.2, To evaluate all models with the same rules, we compared the results only in the Z-X plane as follows.

$$\vec{s}_t = \overline{q}_{t-1}\ \vec{S}_{t+n}\ \overline{q}_{t-1}^{-1}$$
$$Error = \|s_{t+n}(x, z) - w_{t+n}(x, z)\|$$

Contrary to the model proposed by Bremer et al. (2021), our model based on quaternions does not have a special heuristic for short-term predictions. In fact, our implementation is the same, but the time window is $n = 1$.

3.2.2 Data preparation

From now on, and to clarify the explanations, the input dataset created in Bremer's style will be mentioned as LSTM_B, and our adapted version with quaternions will be mentioned as LSTM_Q. To perform the experiments, we also compared them with two previously explained simple algorithms: LE and DES.

Because we have three VR experiences, we can consider that we have three sources of data. To perform the experiments, we followed the same data preparation process for each virtual experience. We also prepared a large data set that mixes all motion data from all virtual experiences together. In this way, the data preparation made from those main datasets to train our models is explained below.

- LE and DES algorithms: as both only need positional original data, only the X-Z positional data have been extracted. In the long- and short-term predictions, the data set is the same since we do not need to prepare the data.
- LSTM_B: In long-term predictions, this model should be fitted with a dataset based on an arithmetic mean follow-

ing a pre-defined time window size of ($n \in \mathbb{Z}$) previous samples to predict the same number of future samples. In this way, the data was prepared following the equations defined in Sect. 2.2, resulting in a new data set with two variables that refer to velocity in the X-Z plane, and two variables that refer to pitch and yaw rotation angles; all under the new reference system explained previously. In the case of the short-term, the data preparation consisted of the same features, but only using the previous sample velocity as a reference system.

- LSTM_Q: The long-term dataset preparation for this model adaptation follows the same rules as the previous one. It accumulates (n) samples as an average rotation, in this case using quaternions, to predict future (n) samples. Following the algorithm mentioned above, this extracted dataset includes the rotational data with four variables, the forward vector with three variables, the velocity vector with three other variables, and the current delta position. All these variables are preprocessed in the new reference system and calculated with the (n) samples collected before the current one. In the case of short-term predictions, this model is exactly the same but using ($n = 1$), then using only the previous quaternion as a reference system.

In the original work, the sample rate was 20Hz, using a $n = 50$ to predict movement in 2.5 s. Because the sample rate in our dataset was recorded at 10Hz, we used only ($n = 25$) to achieve closer results than the original work and make predictions with 2.5 s in the future. For a better understanding, in Fig. 2 is a summary of how this whole experimental process was carried out.

It is worth mentioning that there has not been a normalization process, since all positional data are related to each other. If normalization is applied, the information could be modified in a way that is not predictable. In addition, most of the data follow the same scale in all of the models used.

3.2.3 Experimental setup for locomotion prediction

Taking into account the datasets generated to be used on each method, those have been compared under those conditions. Because LE and DES are simply algorithms, the results are produced directly only by applying the respective equations.

In the case of the DL models based on LSTMs, it is necessary to manage a training procedure. This training procedure will change the inputs and outputs of Bremer's model. In the original model, there are 4 inputs and 2

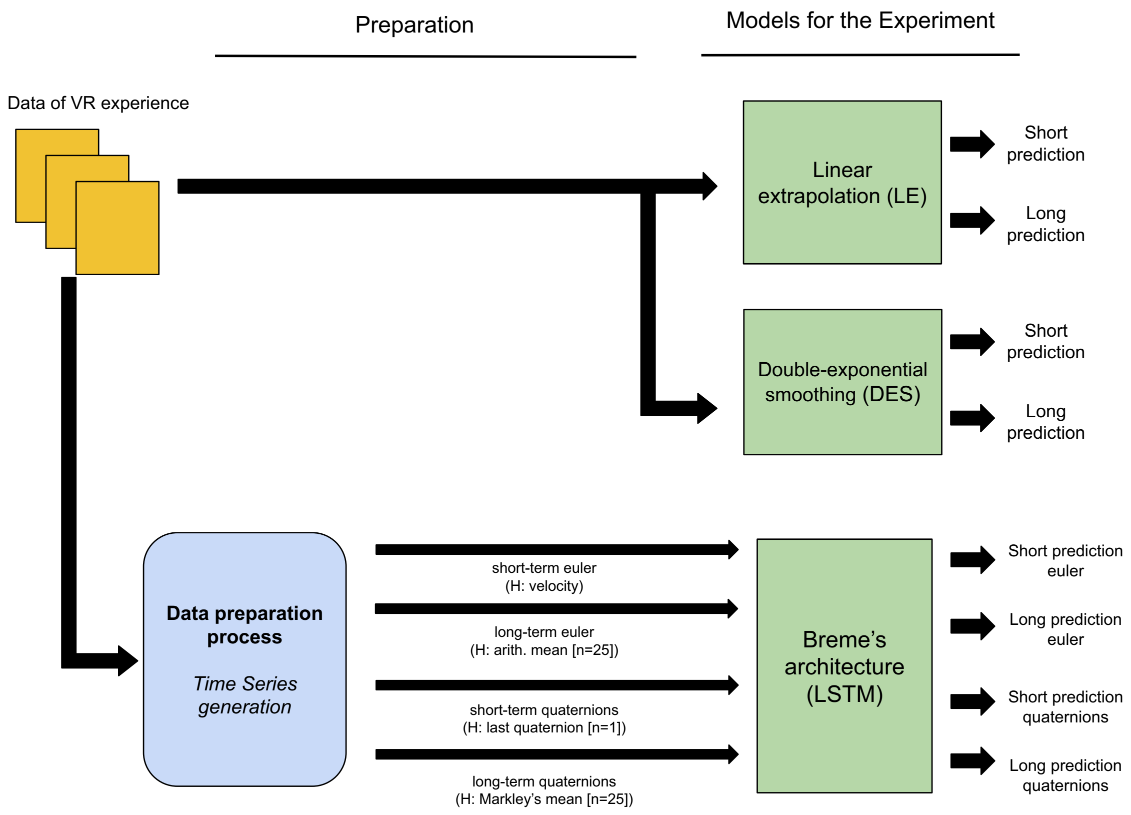
Preparation
Models for the Experiment
Data of VR experience
Linear extrapolation (LE)
Short prediction
Long prediction
Double-exponential smoothing (DES)
Short prediction
Long prediction
Data preparation process
Time Series generation
short-term euler (H: velocity)
long-term euler (H: arith. mean [n=25])
short-term quaternions (H: last quaternion [n=1])
long-term quaternions (H: Markley's mean [n=25])
Breme's architecture (LSTM)
Short prediction euler
Long prediction euler
Short prediction quaternions
Long prediction quaternions

Fig. 2 Overview of the data preparation process and the connection with the different models used in the experiments. The data preparation process has been performed for all experiences and the combination of all of them

outputs when you are not using eye tracking or body sensors. In our adaptation with quaternions, 13 inputs, and 3 outputs are used. Both versions follow the same neural network architecture described in Table 2.

Due to the number of users achieved in our new dataset, and to avoid overfitting, we have reduced the number of epochs with respect to the original experiments made by Bremer et al. (2021). Instead of using 20 as defined in the original paper, only 10 epochs have been used. In each epoch, the training data of the 44 participants are passed to the DL model. With this number of epochs, the error was not further reduced in all experiments.

4 Evaluation

Before starting any evaluation, it is worth remembering that in the DES algorithm, we have two configurable variables (α and β). Because of this, it was necessary to start a grid search process with the DES algorithm and those variables. In this case, the process was applied for both alpha and beta values: 0.1, 0.2, 0.4, 0.6, 0.8, and 1. It should be noted that the value 0 was avoided because if the (α) is equal to 0 it means that no external data are taken into account, making the prediction always the origin of the coordinate system. Since this does not make sense as a prediction method, we avoided that value.

This grid search process was performed to minimize the error in applying these variables. However, the result of this process showed that for both the long-term and short-term, the best value of alpha was equal to 1. Surprisingly, this means that, regardless of the (β) value, the best prediction of DES is equivalent to the prediction of the LE algorithm. In other words, the results are the same but with the addition of unnecessary processing steps.

This is why in the following evaluations, we will see that both algorithms are evaluated together when measuring their MDE.

Table 2 Definition of LSTM architecture used by both RRNs

Layer	Type	Size	Neurons	Dropout
Layer_1	Input embedding	Input-size x 50	–	–
Layer_2	LSTM	50 x 64	64	–
Layer_3	LSTM	50 x 64	64	–
Layer_4	Dropout	50 x 64	–	0.3
Layer_5	Dense	50 x Output-size	Output-size	–

It explains the architecture of this model, where the timestep is 50

4.1 Short-term predictions

After data pre-processing, we fitted the models and algorithms with their own datasets. We can see in Table 3 the results displayed in centimeters of this evaluation in terms of short-term predictions.

As we can see, the quaternions dataset displayed greater results in most of the cases. It is worth mentioning that in the SL virtual scene, this is not true. In this case, the best is the LE algorithm.

Surprisingly, the original dataset which used the velocity of the previous frame as a reference system displayed worse results than the LE algorithm in all scenes. The improvements provided by our quaternion-trained version of the model imply on average 0.52 centimeters more accuracy in terms of error, which is a 27.98% improvement over the previous model. In any case, it should be remembered that the simplest version of Bremer's model, which does not include any additional tracking device, has been used.

4.2 Long-term predictions

Long-term predictions would be different in terms of good prediction achievement. After the training phase, we evaluated all algorithms and DL models with their respective datasets. Results are displayed in Table 4 in the same way as the previous short-term predictions.

The evaluation process displayed completely different results. Here is where the DL models bright more than the algorithmic solutions. In all cases, both the original model and the quaternion adaptation showed much better results.

Table 3 Evaluation results of Short-term datasets measured in centimeters in terms of two dimensions Mean displacement Error (MDE)

Method	Dataset type	SL	ER	SF	All
LE &DES	original	2.105	1.281	1.330	1.497
LSTM	LSTM-B (vel.)	2.852	1.461	1.562	1.874
LSTM	LSTM-Q (n = 1)	2.213	0.992	1.058	1.403

Lower values mean better results (SL = Scene Lab, ER = Escape Room, SF = Shooter Forest, and All = Mixed data sets)

Table 4 Evaluation results of Long-term datasets measured in meters in terms of two dimensions Mean Displacement Error (MDE)

Method	Dataset type	SL	ER	SF	All
LE &DES	original	1.617	0.943	0.875	1.105
LSTM	LSTM-B (n = 25)	1.070	0.512	0.453	0.713
LSTM	LSTM-Q (n = 25)	0.831	0.361	0.313	0.507

Lower values mean better results (SL = Scene Lab, ER = Escape Room, SF = Shooter Forest, and All = All mixed datasets)

Moreover, the quaternion adaptation reduces the error by at least half in all experiences concerning LE. In relation to Bremer's model, the improvements provided by our quaternion-trained version imply on average 0.184 meters more accuracy in terms of error, which is a 27.91% improvement over the previous model.

5 Discussion

Achieving good redirections with RDW methods requires good predictions of where users want to go in the future. Here, we presented a new way to fit deep learning models to achieve better results with this task. To do this, we performed an experiment carried out with 44 participants collecting positional data through three virtual experiences representing common hard-to-track scenarios.

In this way, we obtained a completely new dataset prepared to be used to train DL models. In this way, we tested the data obtained with four different existing prediction algorithms. Those were LE, DES, and the LSTM model presented by Bremer et al. (2021). We adapted Bremer's model using different incoming data pre-preprocessed from our new dataset to obtain greater results than the original work.

Interesting results have also been found during this research process. It is worth mentioning the disappointing results of the DES algorithm. Although it shows itself to be a good algorithmic solution to state-of-the-art motion prediction, its best results have been equivalent to a conventional LE after a grid search procedure. This leads us to believe that the good results of this algorithm in previous work, far from being accurate, are smooth and unchanging. This property can be useful when this algorithm is applied with RDW methods, but they are not good at predicting exactly where the user is going. In other words, we believe that perceptually they are better, but numerically they have worse results when it comes to predicting user locomotion. These findings may be useful for future applications in RDW methods, as they give us clues on how to separate prediction from smoothing (as a perceptual improvement). Being two independent parts within an RDW algorithm.

In terms of the results obtained by the original model presented by Bremer et al. (2021), certain things should be mentioned. First, we avoided using external devices, forcing us to use the simplest form of this model. Second, the sample ratio is slightly slower in our work, consequently doubling the results of the short-term predictions, since they are predicted in 0.1 s and not in 0.05 as in the original work. However, these changes were necessary to compare the algorithm under the new dataset. Despite this, the results for short-term predictions have been shown to be worse than those of the LE algorithm. However, the original work has been quite useful in terms of results and long-term predictions. This must be one of the reasons why the authors continued the work only by analyzing long-term predictions (Stein et al. 2022).

Because favorable results have been achieved by implementing the original LSTM, LE, and DES; we can consider RQ1 answered. This research question was to test whether it would be possible to use the dataset with previous prediction models. From the results obtained, which imply up to a 27% improvement in the predictions, we can confirm that this dataset is suitable for this purpose.

To improve the understanding of how this improved prediction helps RDW methods, we can give an example. The equation describing a curvature gain method in a *steer-to-center* setup (Razzaque 2005) is shown below. It has been extracted from the development by Azmandian et al. (2016) (in the shared Unity project).

$$CG_{t+1} = \frac{||\vec{w}_t||}{C_c} \cdot \sin(\angle(\vec{p}_t, \vec{s}_t))$$

- $\vec{w}_t$ is the delta position vector.
- C_c is a perceptible constant value pre-defined (see Langbehn et al. survey to understand perceptible limits of this constant Langbehn and Steinicke 2018)
- $\vec{p}_t$ is the vector between the user and the center.
- $\vec{s}_t$ is the prediction.
- CG_{t+1} is the resulting curvature gain to be applied in the next frame.

As can be seen in this example, the prediction is a small part of the equation, so future studies are needed to see the true extent of this improved prediction. However, as can be seen in terms of prediction, our model fitting using rotational quaternions and three dimensions has been very favorable for the model. This confirms that it is not necessary to use external devices to improve the results obtained by the original Bremer's model. The results shown appear to be promising both in the long-term and in the short-term. In some cases, it even halves the error compared to the predictions obtained by LE, especially in long-term predictions.

Thanks to the results obtained by this adaptation using quaternions, we can confirm RQ2. This one wanted to test whether our proposed adaptation could obtain better results than previous models or algorithms. This has been affirmative for most of the experiences, both short-term and long-term predictions.

It is interesting to mention the results obtained in general in the SL virtual experience. In all cases and with all models, these have shown the worst results by far. We believe that it is due to the nature of this experience. As explained above, this experience did not have a clear objective. Users simply had to search for a non-existent gnome for a limited time.

We believe that what explains these results is that the nature of this experience caused users to run in order to achieve the objective in time in many cases.

6 Conclusions and future work

Despite the good results obtained, the authors strongly believe that better results could be obtained by changing the model architecture presented by Bremer. We believe there is much room for improvement; for example, using attention modules to identify patterns related to old frames and not just from the immediate before ones. Future work could explore the use of new DL architectures for motion prediction in terms of layers, number of neurons, and input data. Furthermore, the use of eye-tracking devices could be tested in conjunction with quaternions to support claims made in other work with the original model.

Definitely, the results obtained have been promising, and we believe that further research is needed. DL models, in particular, LSTMs, may in the future become part of the RDW methods as an integral part of their operation.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10055-024-00962-9.

Author Contributions All authors are considered to have contributed equally to this manuscript. JMr wrote the main part of the article as the first author and performed the virtual reality experiments. PC and FF analyzed the data as deep learning experts.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. Special thanks to the Programa Propio de la Universidad Politécnica de Madrid granted in July 2023. Financed by the European Union-NextGenerationEU (UP2021-035) through a Margarita Salas contract.

Data availability External data of interest for the reproducibility of the experiments: The dataset can be accessed on the Zenodo platform (Mayor et al. 2023). All Unity projects and Python code could be downloaded from our GitHub repository (Mayor et al. 2023).

Declarations

Necessary statements This section presents the necessary statements about the experiments performed and presented in this manuscript.

Ethical approval Experiments on humans have been performed solely as a means of data collection. No extraneous procedures or experimental devices have been used that could cause harm. According to current legislation, all subjects gave their consent to participate in the experiment and to treat the data anonymously, in no case being able to be identified personally.

Conflict of interest This research does not have a conflict of interest. Funding does not affect the nature of the research. The nature of the research also does not allow for a personal interest that could affect the experiments. All experiments are reproducible, and the data are objective.

References

Azmandian M, Grechkin T, Bolas M, Suma E (2016) The redirected walking toolkit: a unified development platform for exploring large virtual environments. In: 2016 IEEE 2nd workshop on everyday virtual reality (WEVR), pp 9–14. https://doi.org/10.1109/WEVR.2016.7859537

Bradley JV (1958) Complete counterbalancing of immediate sequential effects in a Latin square design. J Am Stat Assoc 53(282):525–528. https://doi.org/10.1080/01621459.1958.10501456

Bremer G, Stein N, Lappe M (2021) Predicting future position from natural walking and eye movements with machine learning. In: 2021 IEEE international conference on artificial intelligence and virtual reality (AIVR), pp 19–28. https://doi.org/10.1109/AIVR52153.2021.00013

Brenneis D, Parker A, Johanson M, Butcher A, Davoodi E, Acker L, Botvinick M, Modayil J, White A, Pilarski P (2021) Assessing human interaction in virtual reality with continually learning prediction agents based on reinforcement learning algorithms: a pilot study

Breuer A, Elflein S, Joseph T, Bolte J-A, Homoceanu S, Fingscheidt T (2019) Analysis of the effect of various input representations for lstm-based trajectory prediction. In: 2019 IEEE intelligent transportation systems conference (ITSC). IEEE, pp 2728–2735

Bölling L, Stein N, Steinicke F, Lappe M (2019) Shrinking circles: adaptation to increased curvature gain in redirected walking. IEEE Trans Vis Comput Gr 25(5):2032–2039. https://doi.org/10.1109/TVCG.2019.2899228

Cho Y-H, Lee D-Y, Lee I-K (2018) Path prediction using lstm network for redirected walking. In: 2018 IEEE conference on virtual reality and 3D user interfaces (VR). IEEE, pp 527–528

Corona E, Pumarola A, Alenya G, Moreno-Noguer F (2020) Context-aware human motion prediction. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 6992–7001

David-John B, Peacock C, Zhang T, Murdison TS, Benko H, Jonker TR (2021) Towards gaze-based prediction of the intent to interact in virtual reality. In: ACM symposium on eye tracking research and applications. ETRA '21 Short Papers. Association for Computing Machinery, New York. https://doi.org/10.1145/3448018.3458008

Dupond S (2019) A thorough review on the current advance of neural network structures. Ann Rev Control 14(14):200–230

Fan C-W, Xu S-Z, Yu P, Zhang F-L, Zhang S-H (2023) Redirected walking based on historical user walking data. In: 2023 IEEE conference virtual reality and 3D user interfaces (VR), pp 53–62. https://doi.org/10.1109/VR55154.2023.00021

Gamage NM, Ishtaweera D, Weigel M, Withana A (2021) So predictable! continuous 3d hand trajectory prediction in virtual reality.

In: The 34th annual ACM symposium on user interface software and technology. UIST '21. Association for Computing Machinery, New York, pp 332–343. https://doi.org/10.1145/3472749.3474753

Grechkin T, Thomas J, Azmandian M, Bolas M, Suma E (2016) Revisiting detection thresholds for redirected walking: combining translation and curvature gains. In: Proceedings of the ACM symposium on applied perception. SAP '16. ACM, New York, pp 113–120. https://doi.org/10.1145/2931002.2931018

Hirt C, Ketzel M, Graf P, Holz C, Kunz A (2022) Short-term path prediction for spontaneous human locomotion in arbitrary virtual spaces. In: 2022 IEEE international symposium on mixed and augmented reality adjunct (ISMAR-adjunct), pp 554–559 (2022). https://doi.org/10.1109/ISMAR-Adjunct57072.2022.00116

Hirt C, Ketzel M, Graf P, Holz C, Kunz A (2022) Heuristic short-term path prediction for spontaneous human locomotion in virtual open spaces. In: 2022 IEEE conference on virtual reality and 3D user interfaces abstracts and workshops (VRW), pp 636–637. https://doi.org/10.1109/VRW55335.2022.00169

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780

Hodgson E, Bachmann E (2013) Comparing four approaches to generalized redirected walking: simulation and live user data. IEEE Trans Vis Comput Gr 19(4):634–643. https://doi.org/10.1109/TVCG.2013.28

Hu Z (2020) Gaze analysis and prediction in virtual reality. In: 2020 IEEE conference on virtual reality and 3D user interfaces abstracts and workshops (VRW), pp 543–544. https://doi.org/10.1109/VRW50115.2020.00123

LaViola JJ (2003) Double exponential smoothing: an alternative to Kalman filter-based predictive tracking. In: Proceedings of the workshop on virtual environments 2003. EGVE '03. Association for Computing Machinery, New York, pp 199–206. https://doi.org/10.1145/769953.769976

Langbehn E, Lubos P, Steinicke F (2018) Evaluation of locomotion techniques for room-scale vr: Joystick, teleportation, and redirected walking. In: Proceedings of the virtual reality international conference (VRIC), Laval. http://basilic.informatik.uni-hamburg.de/Publications/2018/LLS18a

Langbehn E, Steinicke F (2018) In: Lee N (ed) Redirected walking in virtual reality. Springer, Cham, pp 1–11. https://doi.org/10.1007/978-3-319-08234-9_253-1

Lee D-Y, Cho Y-H, Lee I-K (2019) Real-time optimal planning for redirected walking using deep q-learning. In: IEEE conference on virtual reality and 3D user interfaces (VR), pp 63–71. https://doi.org/10.1109/VR.2019.8798121

Lee D-Y, Cho Y-H, Min D-H, Lee I-K Optimal planning for redirected walking based on reinforcement learning in multi-user environment with irregularly shaped physical space. In: 2020 IEEE conference on virtual reality and 3D user interfaces (VR), pp 155–163. https://doi.org/10.1109/VR46266.2020.00034

Markley FL, Cheng Y, Crassidis JL, Oshman Y (2007) Averaging quaternions. J Guid Control Dyn 30(4):1193–1197. https://doi.org/10.2514/1.28949

Mayor J, Calleja P, Fuentes F (2023) Related with: long-short term memory prediction of userś locomotion in virtual reality publication (Dataset). Zenodo. https://doi.org/10.5281/zenodo.8169116

Mayor J, Raya L, Bayona S, Sanchez A (2021) Multi-technique redirected walking method. IEEE Trans Emerg Top Comput 1:1. https://doi.org/10.1109/TETC.2021.3062285

Mayor J, Raya L, Sanchez AA (2019) Comparative study of virtual reality methods of interaction and locomotion based on presence, cybersickness and usability. IEEE Trans Emerg Top Comput 1:1. https://doi.org/10.1109/TETC.2019.2915287

Mayor J, Calleja P, Fuentes F (2023) Github

Nescher T, Kunz A (2013) Using head tracking data for robust short term path prediction of human locomotion. In: Gavrilova ML, Tan CJK, Kuijper A (eds) Transactions on computational science XVIII. Springer, Berlin, pp 172–191

Nescher T, Huang Y-Y, Kunz A (2014) Planning redirection techniques for optimal free walking experience using model predictive control. In: 2014 IEEE symposium on 3D user interfaces (3DUI), pp 111–118. https://doi.org/10.1109/3DUI.2014.6798851

Nescher T, Huang Y-Y, Kunz A (2014) Planning redirection techniques for optimal free walking experience using model predictive control. In: 2014 IEEE symposium on 3D user interfaces (3DUI), pp 111–118. https://doi.org/10.1109/3DUI.2014.6798851

Razzaque S, Kohn Z, Whitton MC (2001) Redirected walking. In: Eurographics 2001—short presentations. Eurographics Association. https://doi.org/10.2312/egs.20011036

Razzaque S (2005) Redirected walking. PhD thesis, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA

Stein N, Bremer G, Lappe M (2022) Eye tracking-based lstm for locomotion prediction in VR. In: 2022 IEEE conference on virtual reality and 3D user interfaces (VR), pp 493–503. https://doi.org/10.1109/VR51125.2022.00069

Stein N (2021) Analyzing visual perception and predicting locomotion using virtual reality and eye tracking. In: 2021 IEEE conference on virtual reality and 3D user interfaces abstracts and workshops (VRW), pp 727–728. https://doi.org/10.1109/VRW52623.2021.00246

Strauss RR, Ramanujan R, Becker A, Peck TC (2020) A steering algorithm for redirected walking using reinforcement learning. IEEE Trans Vis Comput Gr 26(5):1955–1963. https://doi.org/10.1109/TVCG.2020.2973060

Suma EA, Bruder G, Steinicke F, Krum DM, Bolas M (2012) A taxonomy for deploying redirection techniques in immersive virtual environments. In: 2012 IEEE virtual reality workshops (VRW), pp 43–46. https://doi.org/10.1109/VR.2012.6180877

van Rhijn A, van Liere R, Mulder JD (2005) An analysis of orientation prediction and filtering methods for VR/AR. In: IEEE Proceedings of virtual reality, 2005, pp 67–74. https://doi.org/10.1109/VR.2005.1492755

You C, Benda B, Rosenberg ES, Ragan E, Lok B, Thomas J (2022) Strafing gain: redirecting users one diagonal step at a time. In: 2022 IEEE international symposium on mixed and augmented reality (ISMAR), pp 603–611. https://doi.org/10.1109/ISMAR55827.2022.00077

Zank M, Kunz A (2016) Where are you going? Using human locomotion models for target estimation. Vis Comput 32:1323–1335

Zank M, Kunz A (2017) Optimized graph extraction and locomotion prediction for redirected walking. In: 2017 IEEE symposium on 3D user interfaces (3DUI), pp 120–129. https://doi.org/10.1109/3DUI.2017.7893328

Zank M, Kunz A (2015) Using locomotion models for estimating walking targets in immersive virtual environments. In: International Conference on Cyberworlds (CW), pp 229–236. https://doi.org/10.1109/CW.2015.20

Zielasko D, Horn S, Freitag S, Weyers B, Kuhlen TW (2016) Evaluation of hands-free HMD-based navigation techniques for immersive data analysis. In: 2016 IEEE symposium on 3D user interfaces (3DUI), pp 113–119. https://doi.org/10.1109/3DUI.2016.7460040

Zmuda MA, Wonser JL, Bachmann ER, Hodgson E (2013) Optimizing constrained-environment redirected walking instructions using search techniques. IEEE Trans Vis Comput Gr 19(11):1872–1884. https://doi.org/10.1109/TVCG.2013.88