Unscripted Retargeting: Reach Prediction for Haptic Retargeting in Virtual Reality

Aldrich Clarence* Monash University
Jarrod Knibbe† University of Melbourne
Maxime Cordeil‡ Monash University
Michael Wybrow§ Monash University

ABSTRACT

Research is exploring novel ways of adding haptics to VR. One popular technique is haptic retargeting, where real and virtual hands are decoupled to enable the reuse of physical props. However, this technique requires the system to know the users' intended interaction target, or requires additional hardware for prediction. We explore software-based reach prediction as a means of facilitating responsive, unscripted retargeting. We trained a Long Short-Term Memory network on users' reach trajectories to predict intended targets. We achieved an accuracy of 81.1% at approximately 65% of movement. This could enable haptic retargeting during the last 35% of movement. We discuss the implications for possible physical proxy locations.

Index Terms: Human-centered computing—Human computer interaction (HCI)—Interaction paradigms—Virtual reality; Computing methodologies—Machine learning—Machine learning approaches—Neural networks

1 INTRODUCTION

Interacting with objects is a multisensory experience. Currently, consumer virtual reality (VR) devices can provide high resolution visual and audio cues, but only low resolution haptic cues (such as through vibration of controllers). Research in VR has focused on improving these haptic cues through two different approaches: (1) active haptics using specialised devices [2, 15–17, 33], and (2) passive haptics using simple physical objects to represent virtual objects [9, 27]. Active haptic controllers can provide high-resolution haptics for a wide range of objects, but necessitate a complex controller. Though lower resolution, passive haptics have been shown to maintain sense of presence in VR, by supporting object exploration through touching, lifting, and grabbing [26, 46], but they can quickly become impractical as each virtual object requires a physical counterpart.

Haptic retargeting [9] has become a popular approach to passive haptics; enabling the reuse of physical objects to represent multiple virtual objects. By decoupling the user's virtual body from its physical location, the user can synchronously *touch* virtual and real objects in different locations. However, to apply retargeting to a virtual hand, the position of the physical target, virtual target, origin point, and physical hand all need to be known. As a result of this, current approaches [9, 37] mainly implement retargeting in pre-scripted interaction, in which the system specifically assigns which target the user is aiming for.

Previous work [14] has explored solutions for unscripted retargeting by predicting a user's intention. To date, this has relied on additional hardware (e.g., gaze-tracking [14, 20] and specialised interaction surfaces [14]) or additional interactions (e.g., [36]). To support unscripted haptic retargeting, where the user is free to reach for any object they choose, our research explores reach-based target prediction. This prediction can then feed into retargeting techniques, enabling *unscripted retargeting*.

We collect users' reaching behaviour data across 24 targets on a table. We use this data to train a Long Short-Term Memory (LSTM) network to predict the intended target based on initial movement patterns. We examine how early the network can accurately predict the target the user is aiming for, and how this prediction affects the possible placement of a physical object given known detection thresholds of haptic retargeting (e.g., [19, 21, 49]). Across reaches of differing lengths, we achieved an average of 81.1% prediction accuracy at approximately 65% of movement. This could enable haptic retargeting during the last 35% of movement. Our approach and results will enable researchers and developers to implement unscripted retargeting in their applications, helping to enhance haptic feedback in VR without relying on additional specialised devices.

2 RELATED WORK

Haptic Retargeting has emerged as a popular technique for enhancing haptic feedback in VR [9]. Through body warping, haptic retargeting decouples the real and virtual hand, in order to visually reach the virtual object, while actually reaching a spatially-separated physical object. This enables a single physical prop to provide haptic feedback for multiple virtual objects. Recent work has extended haptic retargeting, enabling physical props to represent virtual targets of different weights [42], sizes [11] and shapes [44, 50], as well as to enable bimanual interaction [21].

However, haptic retargeting requires prior knowledge of the location of the virtual object the user is reaching for [9]. As a result of this, haptic retargeting is primarily constrained to scripted interaction, where users are instructed to reach a specific virtual object. We explore movement prediction as a means of allowing users to reach any target they want. We will discuss literature on movement prediction, both for haptic retargeting and more generally across HCI, and retargeting perception thresholds, as these impact the required scope of our prediction.

2.1 Predicting User's Intention

To expand the capability of haptic retargeting beyond scripted scenarios, the users' intended target needs to be known. Sparse Haptic Proxy explores target prediction for retargeting with a specialised hemispherical interaction surface [14]. They predict the users' intended target by examining both gaze and reach dynamics, selecting a target when the user's gaze is fixated on a target for more than a specified time and their hand movement reaches a certain velocity. In their evaluation, they asked users to touch targets presented in a vertical plane in front of them and achieved a 97.5% prediction accuracy. Marwecki et al. [35] also use eye-tracking to understand user's intention and reassign a physical object to multiple virtual objects, although their work does not involve retargeting the user's body, but shifting the virtual object itself. Conversely, Matthews and Smith propose a non-prediction approach, in which users can select the target they want to be redirected to with head gaze [36].

*e-mail: aldrich.clarence@monash.edu
†e-mail: jarrod.knibbe@unimelb.edu.au
‡e-mail: max.cordeil@monash.edu
§e-mail: michael.wybrow@monash.edu

2021 IEEE Virtual Reality and 3D User Interfaces (VR) | 978-1-6654-1838-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/VR50410.2021.00036

2642-5254/21/$31.00 ©2021 IEEE
DOI 10.1109/VR50410.2021.00036

These solutions require either additional hardware or additional interactions. We seek to explore a predictive approach that has no additional hardware requirement, and does not interrupt the user experience by requiring additional interaction.

Prediction is also well studied outside of haptic retargeting with various techniques used to predict the endpoint or future movements of a hand or a cursor. These proposed techniques for prediction can be classified as either target-agnostic or target-aware approaches.

Target-agnostic approaches require no knowledge of the intended targets. Different target-agnostic prediction techniques have been proposed, such as using peak velocity [7], regression-based extrapolation [30, 41], direction of movement and parabola [47], and kinematic template matching [40]. The Kinematic Template Matching approach [40] was extended to be applicable in VR for ray pointing [23], allowing target prediction by comparing the current movement's positional and angular velocity of both head and controller to an existing library of kinematic templates. This enables future predicted movement to be added to the current trajectory, such that potential landing positions can be identified. The technique was able to get prediction accuracy of 3.4° at 90% of the reach. Whilst promising, further work is needed to identify whether this approach can be extended to support non-linear movements (e.g., as needed to reach around other targets or distractor objects) and so be applied to haptic retargeting.

Xia et al. [47] also created a prediction model for touching a target on a touch screen, by modeling users reach trajectories and calculating intersections with the screen. Through evaluation, the system can predict the intended touch location on the screen 128ms before the final touch. We draw inspiration from Xia et al.'s work, to explore target prediction based on reach trajectories in VR.

Unlike target-agnostic approaches, target-aware approaches require the knowledge of the target locations for prediction. Various techniques of target-aware prediction have been proposed, such as calculating the bearing angle to targets [39], using Kalman filters and neural networks [8, 13], inverse optimal control with Bayes' rules [51], Bayesian Inference [3–6], and Gaussian distribution with Bayes' rules [12, 48]. In retargeting, we can rely on target-aware approaches, as we know both the locations of virtual targets and physical proxies. Further, a target-aware approach can benefit from assigning a probability to each target, allowing the application of redirection once the probability of a particular target is high enough, and can ensure that prediction will always align a reach with a target instead of a point in space that has no target.

Recent research has been exploring deep learning approaches to building prediction models for various applications [29, 43]. Biswas et al. [13] used neural networks to predict the cursor's intended target on a computer and achieved more than 90% accuracy at approximately 70% of the reach. In VR, deep learning has been used to enhance the user's experience and interactions by predicting or estimating their behaviour [10, 34]. Makino et al. [34] used past events' motion data to predict the next motion, allowing them to show future body movement in real-time. Even deep learning approaches have been used to optimise the steering algorithm of redirected walking in VR for better experience [31, 32, 45].

Henze et al. [24] also used a neural network to build a predictive model to reduce touch screen latency during continuous touch by predicting the next movement. They showed that the neural network outperforms the extrapolation approach. They further extended their study [25] by using an LSTM network, which is a newer architecture of neural networks, and showed that it performs better than the initial neural network architecture they used in terms of average distance error between actual and predicted touch. LSTM was also used to predict future human activities based on past sequential activities [29], and shown to be able to learn long-term dependencies. With its capability to use sequential data of past events to predict future events, LSTM seems to be a promising approach to predict the user's intention for haptic retargeting. This allows the user to reach any object as we can simply track the user's hand movement, while feeding the data to the LSTM network to predict their intended target and redirect the user's hand towards it. To the best of our knowledge, neural networks have not been used to support unscripted haptic retargeting before.

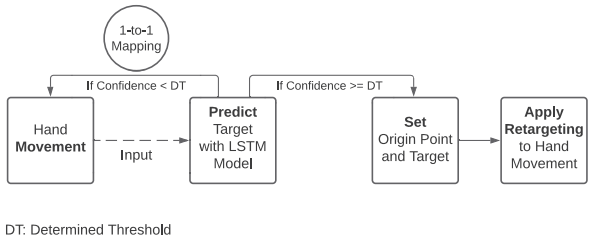


Figure 1: Unscripted haptic retargeting with a prediction model.

2.2 Discrepancy Limits

Haptic retargeting is possible due to visual dominance over proprioception in VR. However, there remains a point at which the user notices a significant mismatch. Such a mismatch might cause some users to feel disturbed during the interaction [49], leading to multiple works investigating the limits of retargeting. The tolerable range to redirect a user's hand was reported to be as large as 40° [14], however Zenner et al. [49] claimed that users can still detect the redirection despite tolerating it. To ensure that redirection is not noticeable, they explored the possible undetected range of retargeting applied to a virtual hand avatar when touching a target located in front of users [49]. They discovered that a user's hand can be redirected horizontally or vertically in a range of 9° and can be scaled up to 6.18% closer or up to 13.75% further. Later, Esmaeili et al. [19] also explored the detection threshold of different scaling applied to the movement of VR controllers on each axis and compound controller movements within a game context. The results show that user's detection of scaling differs for each axis, with the horizontal plane having the lowest threshold and scaling movements in multiple simultaneous axis directions having a higher threshold. They reported that when the controller movement is scaled on compound movements, the movement can be scaled slower by 0.758 and faster by 1.430. Gonzalez et al. [21] further investigated the detection of vertical retargeting when applied in both unimanual and bimanual setups. For a bimanual setup, they explored redirecting the hands in the same and different directions. They discovered there is a larger threshold for retargeting both hands in the same direction than for retargeting the two hands in different directions.

These previous works in detection threshold commonly used a fixed origin point near the user's body as the point to start applying retargeting due to pre-assigning the target the user should reach. In an unscripted scenario, however, the system doesn't know the target to redirect the user's hand towards initially. As the user nears the target, the scope of redirection diminishes. For this reason, we aim to predict the users' intended target as early as possible, and so turn to deep learning prediction approaches. We will be using the previously reported detection thresholds to report how unscripted retargeting affects the possible placement of physical objects given the position at which we can predict a user's intended target.

3 The Unscripted Retargeting Solution

To broaden the current applications of haptic retargeting, we propose Unscripted Retargeting, a software-based prediction solution enabling on-the-fly retargeting. Our predictive approach provides an alternative to the existing on-the-fly retargeting solutions that

require additional eye-tracking hardware or user interactions. Since our approach only requires the motion of the hand (which is already necessary for retargeting), it can be implemented with today's commodity devices (a headset and controllers), and without additional hardware.

To achieve our goal of unscripted retargeting using only hand movements, a prediction model needs to be created. In our approach, a predictive model pre-trained with users' reaches is used to predict the target of a current reach in real-time. We collect users' hand reaching movement from starting position to a set of spatially-separated objects. Following the data collection, data preprocessing is necessary to clean the data and to label the reaches. We can then use this data to create an LSTM-based prediction model that supports early prediction. For evaluation purposes, a subset of the data (not used in the training) can be used for evaluating the prediction model's early prediction performance.

Once the prediction model is pre-trained with users' reaches to a set of spatially-separated virtual objects, it can be used to predict reaches to the same set of virtual objects. As the user begins to move towards their intended target, the system tracks their hand position and predicts the target they are aiming for. Once the system is confident of the predicted target, the current point in space will be set as the origin point and haptic retargeting can be applied, in order for user to visually reach their desired virtual target while physically reaching the closest real object.

4 Data Collection

To create the prediction model, we needed data of users' reaching movements to different locations in a 3D space. We collected data of people's head and hand movements during natural reaches to multiple targets across several repetitions. The collected raw data and our supervised dataset are available online [1]. This data was used to train our prediction model and also evaluate its capability in early prediction. Our data collection was designed to be as close as possible to common implementations of retargeting found in the literature, which typically involved a desktop setup with seated users [9, 20–22]. This allowed direct comparison of our approach with previous works.

4.1 Participants

Twelve participants volunteered to participate in the data collection (6 males, 5 females, and 1 preferred to not answer, with age ranges between 19 to 56 years old, average 32 years old). One participant was left-handed, and all participants had normal or corrected-to-normal vision. When asked about their expertise in Virtual Reality / Augmented Reality technology on a scale of 5-point Likert scale, most participants stated that they were not an expert in VR / AR technology ($Mean$ = 1.92, SD = 1.26). Additionally, they stated how frequently they used VR. Only 2 participants used VR daily, 1 participant used it once a month, and the rest had either never used VR or had used it once or twice before.

4.2 Apparatus

As we intended to explore a software-based unscripted retargeting approach, we only used consumer VR devices and passive objects—an HTC Vive Pro head-mounted display (HMD) with two Vive Trackers, and wooden cylinders. The Vive Base Stations were used to track the positions of the HMD, Trackers, and virtual environment. One Vive tracker was attached at the back of participants' dominant hand with a Velcro strap to track their hand's position and rotation. Another Vive tracker was placed on the table, approximately 52cm away from table's origin point, to calibrate the virtual environment. The table used was approximately 75cm tall, with reach area of 45.5cm by 32.5cm as shown in Figure 2. The reach area contained 12 wooden cylinders with a diameter of 6.5cm and a height of 5.9cm to represent the targets. The distance between the centre of

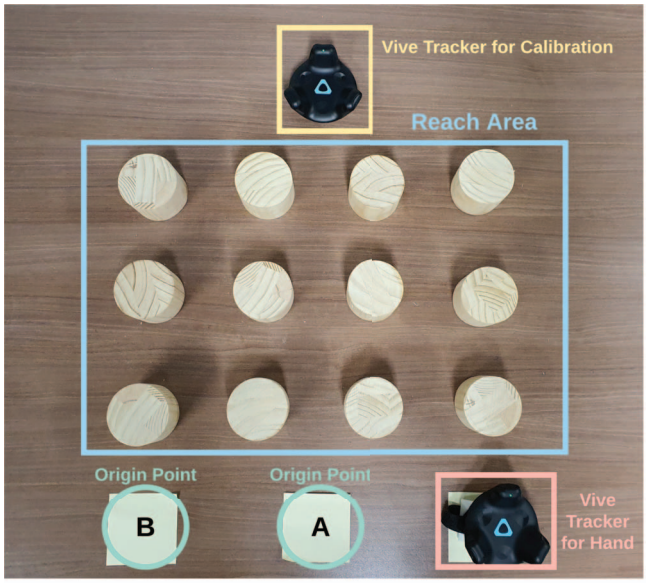


Figure 2: Physical setup of data collection.

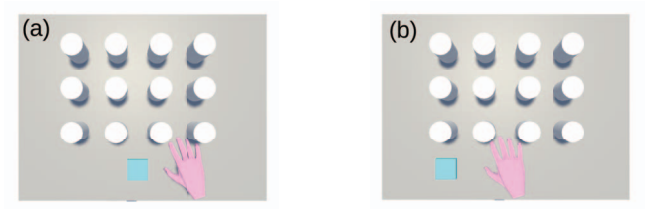


Figure 3: Virtual environment of the two target configurations.

neighbouring cylinders was 13cm, and similar to Zenner et al. [49] we placed the farthest target's centre point 39cm away from the origin point. The design was tested in a pilot study which considered factors like size of cylinders, typical arm span, as well as previous knowledge on limits and application of haptic retargeting [11, 14, 21, 22, 49]. The virtual environment (Figure 3) used to conduct the data collection was developed on Unity3D 2019.1.8f1 and run on a PC.

4.3 Procedure

Participants were given a consent form introducing the data collection procedure. Then, they were asked to answer the pre-task demographic questionnaire.

Participants were then introduced to the tasks they would perform. The researcher helped to attach the Vive Tracker to the participants' dominant hand and helped put on the HMD. Before starting the data collection, each participant performed a short VR-based training session which served to familiarise them with the environment and the data collection task.

After the training, participants conducted the data collection tasks. Inspired by Haptic Retargeting's experiment [9], our data collection task was to reach from an origin point to a target with the palm of their hand and return back to the origin point. Overall in the data collection process, this was repeated 78 times for all conditions per participant. The speed of the reach was not controlled, but the participants were instructed to reach the targets at a natural speed. Natural speed was preferred since retargeting relies on user's visual attention. Overall, the duration of the data collection was approximately twenty-five minutes per participant.

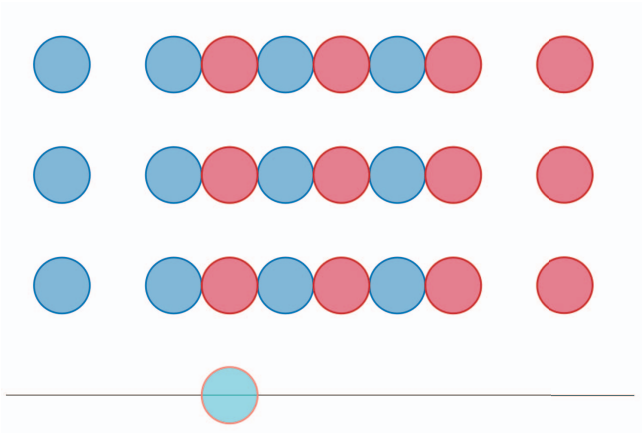


Figure 4: Top-down view when overlaying the two target configurations. Cyan represents the origin point, Blue represents targets in Configuration A, and Red represents targets in Configuration B.

4.4 Design

The data collection process involved collecting users' reaches under different (i) Target Configurations, and (ii) Instruction Types. Before finalising the design, we piloted the tasks to confirm these were appropriate.

There were two target configurations. In the first configuration, users were seated at the front-centre of the targets. In the second configuration, users were seated to the left hand side of the targets. (see Figure 3). In both configurations, the origin point has the same position relative to the participant's body. We included two target configurations for the purpose of increasing the density and coverage of target locations within of our interaction area. The order of the target configurations were counterbalanced.

In each target configuration, participants experienced two instruction types, (a) ordered, and (b) random. In the ordered reaches, participants were required to reach the target that lit up. The order of the targets the participants were instructed to reach followed a Latin Square to account for any potential learning effect. The ordered reaches were then followed by the random reaches where participants were allowed to freely reach any target of their choice from the origin point three times. We explored whether participants reach behaviour differed when they had to decide a target versus being told to reach for a specific target ahead of time.

5 Prediction Model

5.1 Data Preprocessing

After visualising the reaching behaviour of participants (Figure 5c), we were able to assess that reaching behaviours were different between left- and right-handed participants. As a result, we suggest the need for separate prediction models for right-handed and left-handed participants. We initially planned to collect more data for left-handed participants, however, due to data collection restrictions resulting from COVID-19, we decided to exclude the left-handed participant's data and focus on predicting reaches for right-handed people.[1] In addition, we also removed 9 trials from the right-handed participants where there was an issue with the logging or participants misunderstood the instructions. An additional 40 trials were discarded where the duration was twice the standard deviation of the average. The reasons for the increased duration for these 40 removed trials included people making visible mistakes while reaching, deciding the target to reach, or changing their decided target. This left 809 trials for training and evaluation.[2] Through preliminary visual analysis (see Figure 5b), we found no clear differences between the reaches for the ordered and random instruction types. Therefore, we decided to combine both these instruction types together to train and evaluate the prediction model.

We performed data preprocessing to further clean the data and ensure that we train the model only with the data where the participant is actually moving to the target. A reach was considered to begin when the participant's movement first increased to 5% of maximal velocity and considered to end the last time the reach decreased below 5% of maximal velocity, following the same approach as Langavant et al. [18]. Additionally, there were five trials where the system didn't register the initial touch of the target and the participants reached the target multiple times. We manually updated these reaches to end the first time the participant reached the target.

5.1.1 Data Preparation for LSTM

To train the LSTM network, we transformed the data collected into a supervised data set containing the different reaches labelled for a specific target, each comprising a number of features (e.g, hand position x, yaw hand rotation) broken into time steps. Each *observation* represents an instance of a reach (either partial or complete), as shown in Figure 6. Each column represents a single time step of each feature. Each time step contains an average of 10ms of logged data. Due to the variable refresh rate of logging from Unity, each time step in the supervised data set may contain none or multiple logged data events, so we applied linear interpolation for empty time steps. Time taken to complete each recorded reach is different. As a result, we applied post-padding to the time steps after the completed reach using a value of -10 that is outside of the range for each feature, with a masking layer applied in the LSTM to ignore the value.

In order for the LSTM to support prediction of incomplete reaches, we performed data augmentation to the training and validation sets, in which we created multiple instances for each trial with increasingly larger subsets of data, starting from the beginning (see Figure 6).

To build the prediction model, we used the Keras[3] and TensorFlow[4] libraries in Python. We created two LSTM networks. First, we created an LSTM with *hand position* features (x, y, z coordinates) only (LSTM-P). Second, we created an LSTM with both hand *position* features and the hand's yaw *rotation* angle (LSTM-PR). We decided not to use head rotation as a feature, since very little head movement was observed. Probably because all the targets were placed in clear view of users.

To create the prediction model and evaluate its performance, we divided our data into two sets; a training set and an evaluation set. We used stratified 10-fold cross-validation [38] to separate the training and test sets, in which the data in each fold has a similar amount of reaches to each target. Cross-validation allows us to better understand the generalisability of the model. Furthermore, we used a hold-out validation approach, taking 5% of the data from the training set, to ensure our model does not over- or under-fit the data, and to select the epoch with the lowest validation loss.

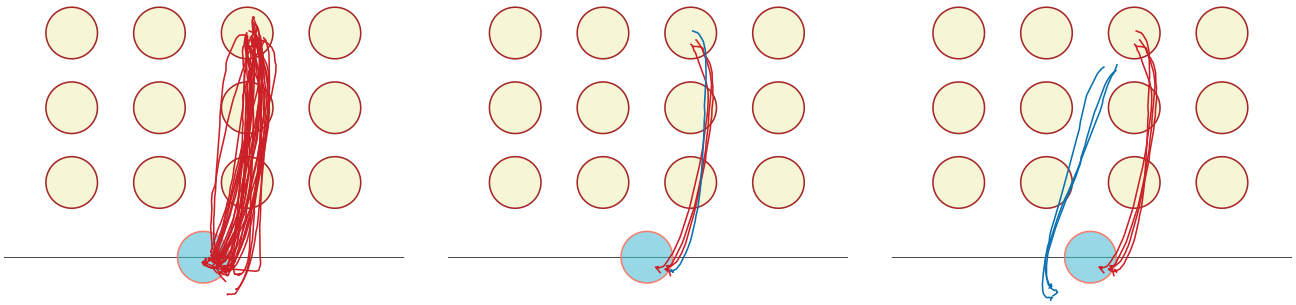
Next, to better understand whether our prediction model can predict reaches by a new user, we also created another variant of the LSTM-P model, using a participant-based split to separate the train and test set (LSTM-P-O). We trained the model with 10 participants' reach data and evaluated the model with another participant's data not used in the training. In total, we created 11 instances of the model to report averaged results.

[1] We expect our approach can be equally used for left-handed prediction given the necessary training data.

[2] We trained another variant of the LSTM-P-O (the model discussed in section 6.2) with the outliers included and found similar results: at 65% of the reach, 68.3% (with outliers) vs 72.3% (without outliers) of the trials were accurately predicted. These results are reported in Figure 8.

[3] Keras: `https://keras.io`

[4] Tensorflow: `https://www.tensorflow.org`



(a) All right-handed participants' reaches.

(b) A single participant's reaches in ordered (red) and random (blue) instruction types.

(c) A comparison between reaches by right-handed (red) and left-handed (blue) participants.

Figure 5: Plots showing examples of tracking data collected from participants reaching to one of the targets.

Feature 1						
Trial	**Instances**	**10ms**	**20ms**	**30ms**	**...**	**500ms**
1	1	0.1	-10	-10	-10	-10
1	2	0.1	0.15	-10	-10	-10
1	3	0.1	0.15	0.2	-10	-10
1	...	....	...	...	...	...
1	50	0.1	0.15	0.2	...	0.9

Figure 6: An example of the LSTM supervised data set structure for a feature (e.g., hand's yaw rotation). The final observation row represents the complete data for that feature for a given reach. We augment the data by creating increasing subsets of this data for each time step, representing partial reaches to this target (the other observation rows).

We used 3 hidden LSTM layers with sizes of 128, 64, and 32 respectively. To avoid model over-fitting, a dropout layer was added with a ratio of 0.2 for LSTM-P and a ratio of 0.4 for LSTM-PR. We used *softmax* as the activation function, categorical cross-entropy, and the Adam algorithm [28] as the optimiser. We set the number of epochs to 30 with a batch size of 32. Similar to Krishna et al. [29], we selected the best performing epoch on the validation set for each model's instance. The selected best model was then evaluated with the test sets.

6 Results

We evaluate the performance of our approach by reporting how well the LSTM prediction models can predict reach targets for users they were trained on and users they were not. We compare the LSTM prediction with several different prediction models in terms of accuracy. Based on the LSTM prediction results, we also present approaches that can be used to determine the retargeting's origin point and show how this translates to the potential placement of physical objects to represent the virtual objects.

6.1 LSTM Prediction Performance

Ten different instances for each of the two LSTM networks, LSTM-P and LSTM-PR, were created using the stratified 10-fold cross-validation. As shown in Figure 9, on average, most targets are accurately predicted at 65% of the reach with the LSTM networks. Due to the use of softmax as the activation function to build our LSTM networks, we can get confidence values; the probability that the current reach is aiming for a certain target. By considering the confidence over time (Figure 7b), the confidence value for the predicted target reaches 85% at approximately 65% of the reach, giving an average error (as seen in Figure 7a) of 2.27cm and 2.26cm for LSTM-P and LSTM-PR, respectively. This confidence level can be used as the pre-determined threshold for when a prediction can be considered as accurate enough to be able to apply retargeting to a user's hand.

Considering the percentage of correctly predicted reaches over time (Figure 8), both LSTM-P and LSTM-PR perform similarly throughout the entire reach with both hitting approximately 81% accuracy at 65% of the reach before peaking at 99.13% and 99.25% accuracy for a full reach with LSTM-P and LSTM-PR, respectively. Based on these results, although LSTM-PR adds yaw rotation as an additional feature to hand's positions in the 3D axis, it performs similar to LSTM-P. This suggests that additional features don't necessarily lead to significantly better prediction performance, and hand position over time is sufficient for predicting reach targets in a desktop setup.

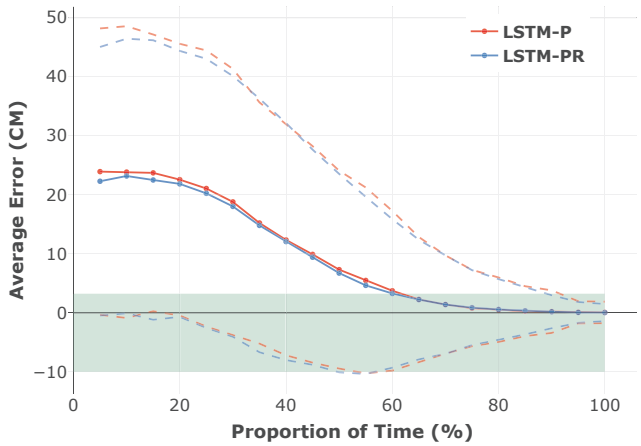
6.2 LSTM Prediction Performance on New Users

In Section 6.1, the LSTM networks were predicting targets from reaches of users whose (other) reaches they had been trained on. We also wanted to evaluate the performance of the LSTM for predicting reaches for people it wasn't trained on. Considering the similar performance between LSTM-P and LSTM-PR, we decided to evaluate just LSTM-P by testing its ability to predict the reaches of a user the model wasn't trained on. Hence, we compared the accuracy and confidence over time between the stratified 10-fold (LSTM-P) and the participant-based cross-validation (LSTM-P-O). The results are shown in Figures 7c, 7d, and 8. Throughout the reach, both cross-validation methods performed similarly, with LSTM-P-O having slightly more error at around 40% to 70% of the reach. Comparing at 65% of the reach, LSTM-P has an error of 2.27cm with an accuracy of 81.1%, and the participant-based split LSTM-P-O has an error of 3.4cm and an accuracy of 72.3%. This suggests that having data of the same user doing the reach in the training set will be beneficial to slightly improve prediction, however, it is not a necessity considering the performance between both cross-validations is very similar.

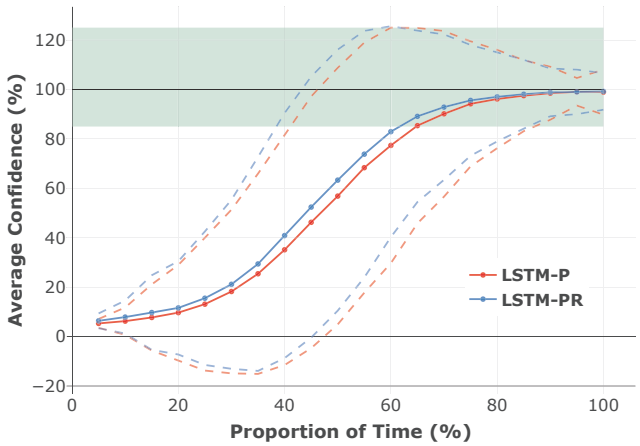
Even at around 70%, the participant-based split LSTM-P-O is already below the average error threshold with an average accuracy of 81.7%. This implies that our approach will support a 'plug-and-play' scenario whereby a prediction model can be pre-trained with other users' reaches and applied to unscripted retargeting to predict new users' reaches.

6.3 Comparison with Other Prediction Models

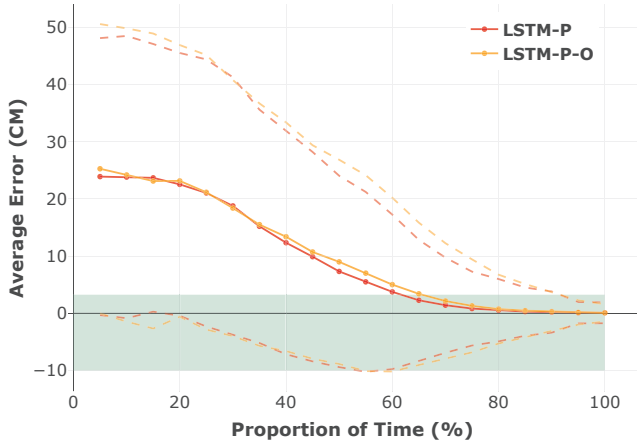
To validate our LSTM networks' performance, we compared our model with existing target-aware prediction approaches in the literature and an alternative approach for prediction.



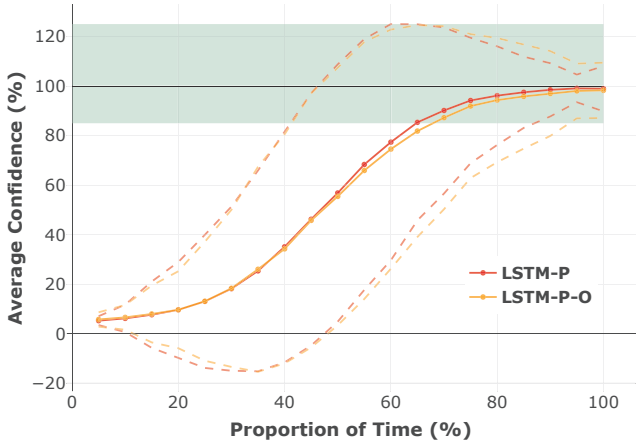
(a) Target Error of LSTM-P and LSTM-PR.



(b) Confidence of LSTM-P and LSTM-PR.



(c) Target Error of LSTM-P and LSTM-P-O.



(d) Confidence of LSTM-P and LSTM-P-O.

Figure 7: The average (a) target error and (b) confidence between different LSTM networks with hand position features only (LSTM-P), and with both hand position features and the hand's yaw rotation angle (LSTM-PR). The average (c) target error and (d) confidence between LSTM-P and a model trained on different users' reaches than the users being evaluated (LSTM-P-O). The dashed lines represent the 2 Standard Deviations above and below the average of each model.

- **Nearest Neighbour.** This served as a baseline. It predicts the closest target from current hand position as the predicted target. This approach was used by Ahmad et al. [4] for mid-air selection on in-car touchscreens.

- **Bearing Angle.** Proposed by Murata [39], this calculates the angle between two vectors, the previous position and the current position, and the current position and the target location, selecting the target with the lowest resultant angle. A buffer of 10 previous frames is used for smoothing.

- **3D Regression.** Finally, we compared our LSTM network to a 3D quadratic regression of hand positions in the training set (x, y, z). The test set's regression was compared to all regressions in the training set, using root mean-squared error (RMSE), with the lowest error selected as the predicted target. Similar to LSTM, we used the same stratified 10-fold cross-validation to split the training and test sets.

To see how each prediction model performs, we plot the average target distance error between actual and predicted target over time at every 5% of the reach in Figure 9. The results showed that the two LSTM networks outperform the other prediction models in terms of accuracy and ability to do early prediction.

With the Nearest Neighbour prediction model serving as a baseline, we can compare how the different prediction models perform in terms of accurately predicting reaches. During the initial 15% of the reach, the LSTM network matches the nearest neighbour's average error, followed by a significant decrease in error compared to the nearest neighbour and other models after 20% of the reach. The nearest neighbour's accuracy catches up with the LSTM networks at 90% of the reach. Since the closest neighbouring target's centre point is 6.5cm away from the real target's centre point, we equate an average error of less than 3.25cm with an accurate prediction. At 65% of the reach, LSTM-P and LSTM-PR perform approximately 5.23x and 5.26x, respectively, better than the baseline in terms of accurately predicting the target. In terms of time, this is on average 1.23x earlier than the nearest neighbour approach.

Comparing the baseline and LSTM networks to the other two models, it seems the 3D regression and bearing angle rely mainly on the direction of movement, and its accuracy will be affected when the neighbouring targets are close to each other or when there are other targets in the same direction of the movement. In general,

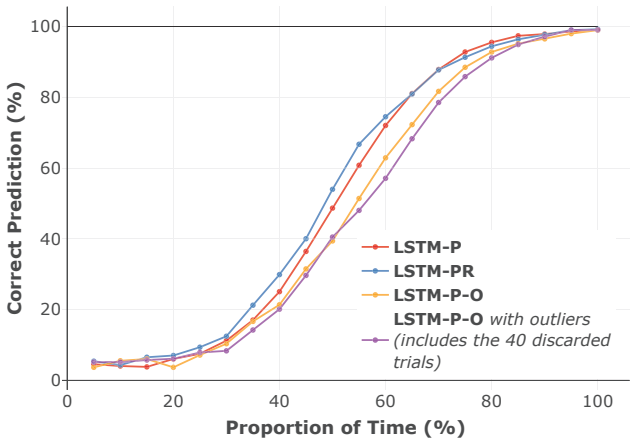


Figure 8: Percentage of correctly predicted trials for different LSTM networks at 5% increments of the reach. All the LSTM networks are trained and evaluated with the 809 trials, except for LSTM-P-O with outliers that includes the 40 discarded trials discussed in section 5.1 (849 trials in total).

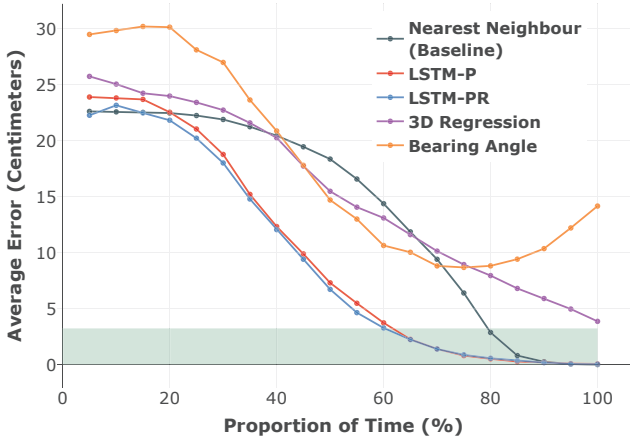
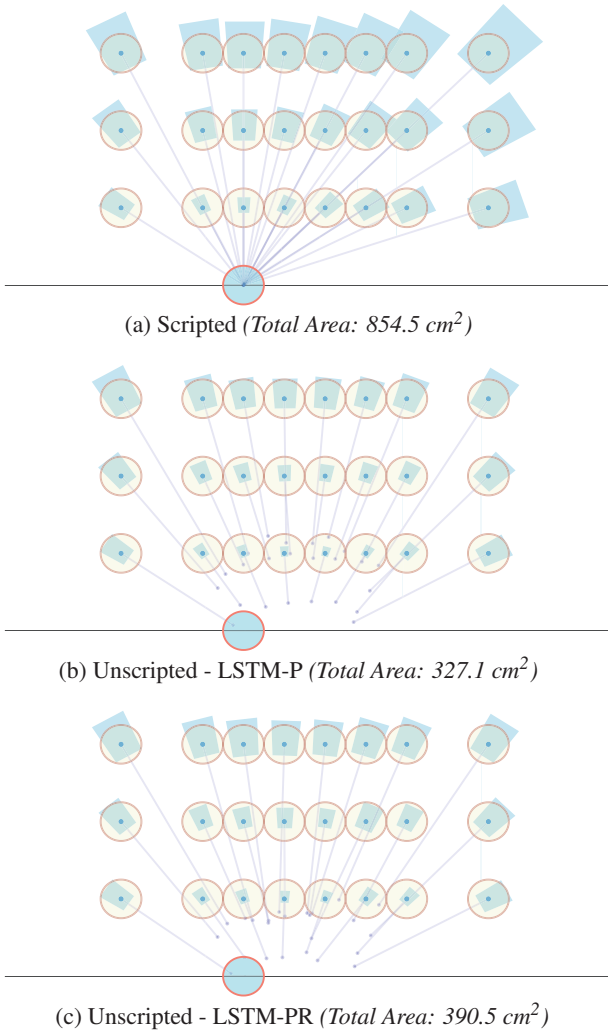


Figure 9: The average distance between the actual and predicted target over time for each prediction model.



(a) Scripted *(Total Area: 854.5 cm^2)*

(b) Unscripted - LSTM-P *(Total Area: 327.1 cm^2)*

(c) Unscripted - LSTM-PR *(Total Area: 390.5 cm^2)*

Figure 10: Scripted vs unscripted retargeting's area based on the threshold of Zenner et al. [49]. The blue region of each virtual target shows the possible locations to place the centre point of the physical counterpart. The points at the beginning of the rays represent the average point the model predicts the target user is reaching with confidence higher than the pre-determined threshold (85%).

the nearest neighbour approach performs better than 3D Regression and Bearing angle throughout the reach, except during the middle of the reach. 3D regression manages to perform better over time, however, it can't distinguish close targets as well, especially when a reach's motion is similar to the neighbouring target, resulting in worse performance than the nearest neighbour. Bearing angle assumes movement to a path is straight and previous works [8, 39] use it in an environment where there are no targets in the same linear direction. It is apparent that when bearing angle is applied to an environment with targets in the same direction, it doesn't perform as well as previous works have shown. The worse performance after 75% of the reach is probably caused by an assumption that the reach is not completed since there are targets further in the same direction.

6.4 Undetectable Retargeting Area

In scripted retargeting, the origin point is placed near the body. Azmandian et al. [9] have a home button placed in front of users, close to the body. For unscripted retargeting, the origin point placement is dynamic and dependent on the point at which the target can be predicted. To demonstrate how prediction supports unscripted retargeting, we compare the possible proxy area for scripted and unscripted retargeting. This is the area where physical objects can be placed to represent virtual objects, conforming with the published thresholds at which retargeting remains below human perception thresholds. With the distraction-less detection threshold limit proposed by Zenner et al. [49], we computed the possible retargeting area for each target, based on the average point from which we can accurately predict each target.

Figure 10 shows the retargeting area (i.e., the possible locations of the centre point for a physical object to be placed at to represent each virtual object) for scripted retargeting and the unscripted approaches with LSTM-P and LSTM-PR. We find that the area of scripted retargeting is 2.6x and 2.2x larger than LSTM-P and LSTM-PR, respectively. Based on the retargeting area overlap of all 24 virtual objects in our setup, all 24 physical counterparts are necessary for unscripted retargeting, whereas scripted retargeting requires only 22 physical objects, assuming that the touch needs to end up at the centre point of the object.

7 Discussion

Our results show that unscripted retargeting is possible utilising data collected from consumer VR hardware with a LSTM-based prediction model.

Scripted retargeting, where users are told which objects to reach for, enables each physical proxy to provide haptic feedback for the widest range of virtual objects. However, this constrains the users' ability to select objects on-the-fly. Using our LSTM prediction model to support dynamic target selection across a tabletop surface of 45 x 32cm, the total retargeting area of our 24 targets is 54.3% smaller than when applying scripted retargeting, below perceptive threshold. However, this translates to requiring only two additional physical props to support unscripted interaction, assuming the touch should end up at the centre point of a physical proxy. We believe this is a conservative estimate, as Zenner et al.'s thresholds are derived through a slightly different study design, and that further work on perception thresholds for tabletop-scale interaction and size of the target may further support the applicability of prediction approaches.

Looking at the retargeting area (Figure 10) when applying the proposed detection threshold by Zenner et al. [49], a passive physical object on its own cannot represent a broad space of virtual objects, if intended to be unnoticeable. When a target is close to the starting hand position, it is practically impossible to apply retargeting undetected, as movement distance is required to spatially separate the real and virtual hands. This leads to alternative solutions for better physical objects 'reusability', such as using hybrid warping, where decoupling is applied to both the environment and the body [9], and combining retargeting with active haptics [20]. Our work may further support the enhancement and reusability of these approaches too, but this remains an avenue for future work.

Our paper focuses on exploring the different software-based approaches to early prediction, allowing for unscripted interaction in retargeting. Current solutions to support an unscripted interaction in retargeting require additional hardware like eye-tracking and a specialised proxy [14] that are not common in consumer VR devices yet, or interactions to indicate targets prior to reach [36] that could detract from the overall user experience. While our prediction model is presented as an alternative approach to unscripted retargeting, further work should explore how our work can combine with existing eye-tracking prediction for a better overall prediction performance across the virtual environment, including differentiating distant and close objects.

Our exploration closely followed the setup proposed by Azmandian et al. [9], using both a tabletop space and reaches departing from a specific location. We focused on predicting discrete targets (i.e., picking a target from a known set). In this case, we need training data for each target. This general approach, using an LSTM for early prediction, would generalise across different spatial setups and designs (i.e., different object layouts and shapes) provided it was trained with new data. Further, we achieve accurate prediction results using only the hand's tracked position (LSTM-P). This helps our approach to be hand-pose invariant, which may come to support generalisation across different object shapes. However, in the future, as spaces become more complex or more continuous prediction is desired (i.e., starting from any location, or predicting a coordinate in space), it may become necessary to include additional tracking features that we don't train the model on, such as head angle and hand rotation. This remains an avenue for future work.

Due to the challenges of COVID-19, we were unable to validate our prediction model in a user study. Instead, we used participant-based cross-validation, to examine our model's performance on data from an unseen user. Through this, we were still able to achieve 72.3% correctly predicted reaches at 65% of the reach, making us confident of our model's performance. Combining this prediction with user experience evaluations remains another exciting avenue for future work.

Finally, the pre-trained LSTM network stores only the weights of each neuron to map input to output, and is therefore able to predict data quickly in real-time. While our LSTM exploration was intended for retargeting, it also suggests potential of LSTM prediction for other purposes such as reducing latency of graphical feedback [47] due to its ability to predict user's intention early.

8 Conclusion

Haptic Retargeting has shown that user's real and virtual hand can be decoupled to reuse a physical object as a haptic feedback for multiple virtual objects, when a target is known by the system. We broaden the application of retargeting by proposing a software-based prediction model using LSTM that allows retargeting to support unscripted interaction. We demonstrated the potential of our solution to apply unscripted retargeting that is applicable to current consumer VR devices. Our results show that a neural network prediction model pre-trained with data of hand positions and rotation can be integrated with haptic retargeting to allow unscripted reach interaction. In this model, a user can reach a target normally without needing a pre-assigned target, and retargeting is applied once the model is able to predict the target the user is aiming for. We showed how our proposed LSTM prediction models performed favourably against other target-aware prediction approaches in a spatially-dense setup. With LSTM, prediction with an average accuracy of 81.1% can be achieved at around 65% of the reach. Our findings suggests LSTM is a good choice for a prediction model for unscripted retargeting due to its ability to do early prediction 1.23x earlier than using the nearest neighbour approach. We also explored how this prediction model translates to the possible haptic retargeting area, where physical objects could likely be placed without the retargeting being detected. Our findings suggest that with a desktop setup, integrating our prediction model will require 1.09x more physical objects than scripted retargeting. Our research demonstrates an approach for providing a better haptic experience in future VR applications via unscripted retargeting using only common VR devices.

References

[1] Reaching Dataset of Unscripted Retargeting. https://doi.org/10.26180/13615868, 2021. [Online; accessed 22-January-2021].

[2] P. Abtahi and S. Follmer. Visuo-haptic illusions for improving the perceived performance of shape displays. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pp. 1–13, 2018.

[3] B. I. Ahmad, C. Hare, H. Singh, A. Shabani, B. Lindsay, L. Skrypchuk, P. Langdon, and S. Godsill. Selection facilitation schemes for predictive touch with mid-air pointing gestures in automotive displays. In *Proceedings of the 10th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, pp. 21–32, 2018.

[4] B. I. Ahmad, P. M. Langdon, S. J. Godsill, R. Donkor, R. Wilde, and L. Skrypchuk. You do not have to touch to select: A study on predictive in-car touchscreen with mid-air selection. In *Proceedings of the 8th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, pp. 113–120, 2016.

[5] B. I. Ahmad, P. M. Langdon, S. J. Godsill, R. Hardy, E. Dias, and L. Skrypchuk. Interactive displays in vehicles: Improving usability with a pointing gesture tracker and bayesian intent predictors. In *Proceedings of the 6th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, pp. 1–8, 2014.

[6] B. I. Ahmad, J. K. Murphy, P. M. Langdon, S. J. Godsill, R. Hardy, and L. Skrypchuk. Intent inference for hand pointing gesture-based interactions in vehicles. *IEEE transactions on cybernetics*, 46(4):878–889, 2015.

[7] T. Asano, E. Sharlin, Y. Kitamura, K. Takashima, and F. Kishino. Predictive interaction using the delphian desktop. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, pp. 133–141, 2005.

[8] G. A. Aydemir, P. M. Langdon, and S. Godsill. User target intention recognition from cursor position using kalman filter. In *International*

Conference on Universal Access in Human-Computer Interaction, pp. 419–426. Springer, 2013.

[9] M. Azmandian, M. Hancock, H. Benko, E. Ofek, and A. D. Wilson. Haptic retargeting: Dynamic repurposing of passive haptics for enhanced virtual reality experiences. In *Proceedings of the 2016 chi conference on human factors in computing systems*, pp. 1968–1979, 2016.

[10] Y. Ban. Estimating the direction of force applied to the grasped object using the surface emg. In *International Conference on Human Haptic Sensing and Touch Enabled Computer Applications*, pp. 226–238. Springer, 2018.

[11] J. Bergström, A. Mottelson, and J. Knibbe. Resized grasping in vr: Estimating thresholds for object discrimination. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, pp. 1175–1183, 2019.

[12] X. Bi and S. Zhai. Bayesian touch: a statistical criterion of target selection with finger touch. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*, pp. 51–60, 2013.

[13] P. Biswas, G. A. Aydemir, P. Langdon, and S. Godsill. Intent recognition using neural networks and kalman filters. In *International Workshop on Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data*, pp. 112–123. Springer, 2013.

[14] L.-P. Cheng, E. Ofek, C. Holz, H. Benko, and A. D. Wilson. Sparse haptic proxy: Touch feedback in virtual environments using a general passive prop. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pp. 3718–3728, 2017.

[15] I. Choi, H. Culbertson, M. R. Miller, A. Olwal, and S. Follmer. Grabity: A wearable haptic interface for simulating weight and grasping in virtual reality. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, pp. 119–130, 2017.

[16] I. Choi, E. W. Hawkes, D. L. Christensen, C. J. Ploch, and S. Follmer. Wolverine: A wearable haptic interface for grasping in virtual reality. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 986–993. IEEE, 2016.

[17] I. Choi, E. Ofek, H. Benko, M. Sinclair, and C. Holz. Claw: A multifunctional handheld haptic controller for grasping, touching, and triggering in virtual reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pp. 1–13, 2018.

[18] L. C. De Langavant, P. Remy, I. Trinkler, J. McIntyre, E. Dupoux, A. Berthoz, and A.-C. Bachoud-Levi. Behavioral and neural correlates of communication via pointing. *PLoS One*, 6(3), 2011.

[19] S. Esmaeili, B. Benda, and E. D. Ragan. Detection of scaled hand interactions in virtual reality: The effects of motion direction and task complexity. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 453–462. IEEE, 2020.

[20] E. J. Gonzalez, P. Abtahi, and S. Follmer. Reach+ extending the reachability of encountered-type haptics devices through dynamic redirection in vr. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, pp. 236–248, 2020.

[21] E. J. Gonzalez and S. Follmer. Investigating the detection of bimanual haptic retargeting in virtual reality. In *25th ACM Symposium on Virtual Reality Software and Technology*, pp. 1–5, 2019.

[22] D. T. Han, M. Suhail, and E. D. Ragan. Evaluating remapped physical reach for hand interactions with passive haptics in virtual reality. *IEEE transactions on visualization and computer graphics*, 24(4):1467–1476, 2018.

[23] R. Henrikson, T. Grossman, S. Trowbridge, D. Wigdor, and H. Benko. Head-coupled kinematic template matching: A prediction model for ray pointing in vr. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–14, 2020.

[24] N. Henze, M. Funk, and A. S. Shirazi. Software-reduced touchscreen latency. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*, pp. 434–441, 2016.

[25] N. Henze, S. Mayer, H. V. Le, and V. Schwind. Improving software-reduced touchscreen latency. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services*, pp. 1–8, 2017.

[26] H. G. Hoffman. Physically touching virtual objects using tactile augmentation enhances the realism of virtual environments. In *Proceedings. IEEE 1998 Virtual Reality Annual International Symposium (Cat. No. 98CB36180)*, pp. 59–63. IEEE, 1998.

[27] B. E. Insko, M. Meehan, M. Whitton, and F. Brooks. *Passive haptics significantly enhances virtual environments*. PhD thesis, University of North Carolina at Chapel Hill, 2001.

[28] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[29] K. Krishna, D. Jain, S. V. Mehta, and S. Choudhary. An lstm based system for prediction of human activities with durations. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(4):1–31, 2018.

[30] E. Lank, Y.-C. N. Cheng, and J. Ruiz. Endpoint prediction using motion kinematics. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pp. 637–646, 2007.

[31] D.-Y. Lee, Y.-H. Cho, and I.-K. Lee. Real-time optimal planning for redirected walking using deep q-learning. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 63–71. IEEE, 2019.

[32] D.-Y. Lee, Y.-H. Cho, D.-H. Min, and I.-K. Lee. 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. IEEE, 2020.

[33] J. Lee, M. Sinclair, M. Gonzalez-Franco, E. Ofek, and C. Holz. Torc: A virtual reality controller for in-hand high-dexterity finger interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–13, 2019.

[34] Y. Makino, Y. Shioi, Y. Horiuchi, and H. Shinoda. Interference of projected future self. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pp. 4113–4117. IEEE, 2019.

[35] S. Marwecki, A. D. Wilson, E. Ofek, M. Gonzalez Franco, and C. Holz. Mise-unseen: Using eye tracking to hide virtual reality scene changes in plain sight. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, pp. 777–789, 2019.

[36] B. J. Matthews and R. T. Smith. Head gaze target selection for redirected interaction. In *SIGGRAPH Asia 2019 XR*, pp. 13–14. 2019.

[37] B. J. Matthews, B. H. Thomas, S. Von Itzstein, and R. T. Smith. Remapped physical-virtual interfaces with bimanual haptic retargeting. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 19–27. IEEE, 2019.

[38] F. Mosteller and J. W. Tukey. Data analysis, including statistics. *Handbook of social psychology*, 2:80–203, 1968.

[39] A. Murata. Improvement of pointing time by predicting targets in pointing with a pc mouse. *International Journal of Human-Computer Interaction*, 10(1):23–32, 1998.

[40] P. T. Pasqual and J. O. Wobbrock. Mouse pointing endpoint prediction using kinematic template matching. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 743–752, 2014.

[41] J. Ruiz and E. Lank. Effects of target size and distance on kinematic endpoint prediction. Technical report, Citeseer, 2009.

[42] M. Samad, E. Gatti, A. Hermes, H. Benko, and C. Parise. Pseudo-haptic weight: Changing the perceived weight of virtual objects by manipulating control-display ratio. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–13, 2019.

[43] Z. Shi, M. Xu, Q. Pan, B. Yan, and H. Zhang. Lstm-based flight trajectory prediction. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2018.

[44] J. Spillmann, S. Tuchschmid, and M. Harders. Adaptive space warping to enhance passive haptics in an arthroscopy surgical simulator. *IEEE transactions on visualization and computer graphics*, 19(4):626–633, 2013.

[45] R. R. Strauss, R. Ramanujan, A. Becker, and T. C. Peck. A steering algorithm for redirected walking using reinforcement learning. *IEEE Transactions on Visualization and Computer Graphics*, 26(5):1955–1963, 2020.

[46] B. G. Witmer and M. J. Singer. Measuring presence in virtual environments: A presence questionnaire. *Presence*, 7(3):225–240, 1998.

[47] H. Xia, R. Jota, B. McCanny, Z. Yu, C. Forlines, K. Singh, and D. Wigdor. Zero-latency tapping: using hover information to predict touch

locations and eliminate touchdown latency. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pp. 205–214, 2014.

[48] D. Yu, H.-N. Liang, X. Lu, K. Fan, and B. Ens. Modeling endpoint distribution of pointing selection tasks in virtual reality environments. *ACM Transactions on Graphics (TOG)*, 38(6):1–13, 2019.

[49] A. Zenner and A. Krüqer. Estimating detection thresholds for desktop-scale hand redirection in virtual reality. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 47–55. IEEE, 2019.

[50] Y. Zhao and S. Follmer. A functional optimization based approach for continuous 3d retargeted touch of arbitrary, complex boundaries in haptic virtual reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2018.

[51] B. Ziebart, A. Dey, and J. A. Bagnell. Probabilistic pointing target prediction via inverse optimal control. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*, pp. 1–10, 2012.