Towards Predicting VR Waist Location from Sparse Tracking Data

Samuel Li, Uzair Mirza, Cheuk Ho Yun

Abstract

We present a neural architecture for predicting the waist location of a virtual reality (VR) user, given only the position and orientation data for the headset and hand controllers. Such a model could offer a more immersive user experience without requiring additional tracking hardware, lowering the barrier to entry for general VR adoption. We reproduce the LSTM architecture from [1], and show that our proposed modifications yield a significant reduction of the root-mean-square error in the predicted waist position and orientation.

1 Introduction

A common problem in virtual reality (VR) software development is the lack of the positional information on the users' untracked body parts. An entry-level VR system is usually shipped with one head-mounted display (HMD) and two hand-held controllers, each of which have their position and rotation tracked using built-in gyroscopes, cameras or laser trackers. The head and the hands' locational information is therefore available to the devices. However, the other parts of the user's body are not tracked and therefore cannot be utilized in VR software.

The waist is generally not tracked directly. However, the waist is one of the more important body parts to track, since:

- The waist's position and rotation encodes important information about the player's posture. Imagine you are playing an online multiplayer game, you are hiding behind a wall and you are peeking at your opponent by leaning your head to one side. Your virtual body, if rendered accurately, should be hidden behind the wall. However, with only the head and the hands' locational information, the game will very likely render your body at an inaccurate position, exposing your body to your opponent.
- The waist area is often used as an "inventory" in games. Pistols and other small items are often stored on the "virtual belt" of the player, and it is important that the player can reach them reliably.

The most common approach to rendering the waist of a player is to place the waist where the head is, with a fixed vertical offset. This approach obviously fails to meet the aforementioned requirements. We therefore attempt to solve the problem by training a model that aims to predict accurately the position and the rotation of the player's waist, given the locational data of their head and hands.

1.1 Related Work

In [1], Jothi et. al. propose an LSTM-based architecture for solving the waist positioning problem. Their architecture consists of two LSTM layers of 512 and 128 units, respectively, with a fully connected 3-unit output layer and intermediate dropout layers. This model is itself a refinement of the model presented in [2] by Lee et. al. We also add a second 4-unit output layer to predict the waist rotation quaternion. [In the original model, both outputs were combined into a single 7-output layer.]

We reproduce the results from this work (with a different dataset), and improve on their results with a modified model.

In [2], Lee et. al. present a neural architecture for predicting the waist orientation using only position and orientation data of a head-mounted device. Their model has a discretized output space (left, right, or forward), and achieves 90% classification accuracy. Our results are not directly comparable since our model outputs a full quaternion rather than a discretized one-hot classification. However, our final results (Table 1) are competitive with the effective accuracy of [2], given the angular size of each output class.

2 Design & Architecture

2.1 Data Generation

Since the target of the model is to predict a VR user's waist position, the most direct way of collecting annotated training data is to have a person wear a VR headset with two controllers, using an additional tracker to track their waist's locational data while performing movement.

There are, however, some difficulties in the implementation of this method. To produce accurate real-time waist data, we would need a SteamVR-compatible tracker, which is costly ($\sim$\$500). Furthermore, recording a large dataset of different motions that is representative of an average VR user's movements is very time-consuming.

We therefore decided to simulate a set of motions using existing motion capture data. We place a humanoid character in a scene built with Unity Engine, attach objects to the humanoid's head, hands and waist, and track their locational information (position and rotation) at a fixed 150 Hz framerate while animation clips are played on this character. The animation clips are downloaded from Mixamo, a royalty-free online animation database provided by Adobe.

The animation clips are chosen with these specifications:

- We avoided unrealistic movements. For example, a VR user is very unlikely to be proning on the floor or doing a back flip. Movements such as combating with a sword or drawing an arrow from a quiver on the back, on the other hand, make a lot more sense.
- The waist's position is especially hard to predict using naive algorithms when the player is looking downwards or crouching. We made sure that around half of the clips involve these movements.

Once the animation clips are chosen, we track the rotation (a quaternion composed of 4 floats) and position (a Vector3 composed of 3 floats) for each of the four body parts (head, left hand, right hand, waist). For the head, the only dimension of its positional data goes into the model is the height, because the head's location in the x-z plane does not encode any information of the player's posture and may introduce unwanted noise. The hands' and the waist's position and rotation are measured relative to the head.

2.2 Data Augmentation

The dataset is broken into 63 individual motions (e.g. standing, walking), each one of which lasts 2-5 seconds. We reserve 5 motions for validation, 5 for test data, and use the remaining motions as training data. We train on an infinite data generator that performs the following data augmentation:

- Since quaternions form a double-cover of the rotation group, we randomly perform a global negation of the rotation quaternions for the head, left controller, and right controller. This amplifies the data by a factor of 8.
- We globally mirror the position and rotation data with 50% probability. This amplifies the data by a further factor of 2.
- We globally rotate the character by an arbitrary angle along the vertical axis. This only affects the head rotation quaternion, since we represent all other rotations as quaternions relative to the head orientation.
- We uniformly scale the entire motion by up to 20% about the player's feet.

- We slow down or speed up the motion by up to 20% via linear interpolation.

We then randomly extract a 128-frame ($\sim 0.85\,\mathrm{s}$) window of the selected motion to serve as a training example.

2.3 Reproduction of Jothi et. al.

We reproduced the architecture presented in [1] and trained the model on our self-generated dataset. We use the Adam optimizer with a learning rate of 10^{-3} and a batch size of 32. Since our data is augmented and streamed from an infinite generator, we train for 10000 batches rather than a fixed number of epochs.

We use a mean-squared error loss function for the waist position output, as this directly corresponds to the RMS positioning error. For the waist rotation quaternion, we cannot use the mean-squared error, since the quaternions q and $-q$ correspond to the same physical orientation. Instead, we use the loss function

$$\mathcal{L}(q, q_0) := -\langle q, q_0\rangle^2 + \kappa(q^2 - 1)^2,$$

where q, q_0 are the predicted and target quaternion, respectively, and $\kappa \gg 1$ is used to constrain the output quaternion to the unit 3-sphere. We empirically found that $\kappa = 100$ gives good results.

The loss function is minimized when q, q_0 are maximally aligned or antialigned. In fact, given two unit quaternions q, q' representing physical rotations R, R', the minimum angle θ required to transform between R and R' satisfies the formula

$$\cos\theta = 2\langle q, q'\rangle^2 - 1.$$

Thus our loss function encodes the physical distance between orientations in $\mathrm{SO}(3)$.

The RMS waist positioning error (proportional to the square root of the MSE loss) during training is shown in Figure 3. The RMS rotation error (degrees) is shown in Figure 4. We successfully reproduce the $10\,\mathrm{cm}$ RMS positioning accuracy reported in [1].

Note that [1] did not use any form of data augmentation, and trained with mean-squared-error loss only. The corresponding training curves without any data augmentation are shown in Figure 5 and 6. As evident from the training curves, the data augmentation substantially decreases overfitting. Moreover, the augmentation leads to a significant improvement in test performance, as shown in Table 1. The data augmentation also appears to stabilize the training, since it is less likely that a batch will contain nearly-identical training sequences by chance (this causes the spikes in Figures 5 and 6).

2.4 Proposed Model

We propose several changes to the model from [1]:

- Unlike in text classification tasks, there is likely no significant benefit to retaining long-term context when predicting waist position and orientation. Only short-term context is needed to maintain temporal coherence. Thus we propose replacing the LSTM layers with GRU (gated recurrent unit) layers.
- The quaternion output is constrained to a sphere, which is a nonlinear output manifold. The network may benefit from additional nonlinearity for this output, freeing up additional LSTM units for the position prediction. Thus we propose adding a 64-unit ReLU hidden layer between the 4-unit quaternion output and the 128-unit LSTM layer.
- The quaternion output cannot have any values outside of the range $[-1, 1]$. Thus we add a softsign activation function to the 4-unit quaternion output. [We use softsign rather than tanh to avoid the vanishing gradient problem.]

As shown in Table 1, these modifications indeed improve the validation performance.

3 Results

Table 1 shows the position and rotation prediction performance of each model, with and without data augmentation, on the test set. We see that our data augmentation procedure and proposed modifications lead to a clear improvement in RMS waist tracking error.

Model	Tracking Error (cm)	Orientation Error (degrees)
Reproduction of [1], with DA	8.02	22.39
Reproduction of [1], without DA	12.04	27.35
Proposed, with DA	**6.38**	**20.47**
Proposed, without DA	11.95	27.54

Table 1: RMS waist tracking error and RMS orientation error for each of the trained models, evaluated over the test set. Results are shown with data augmentation (DA) enabled and disabled.

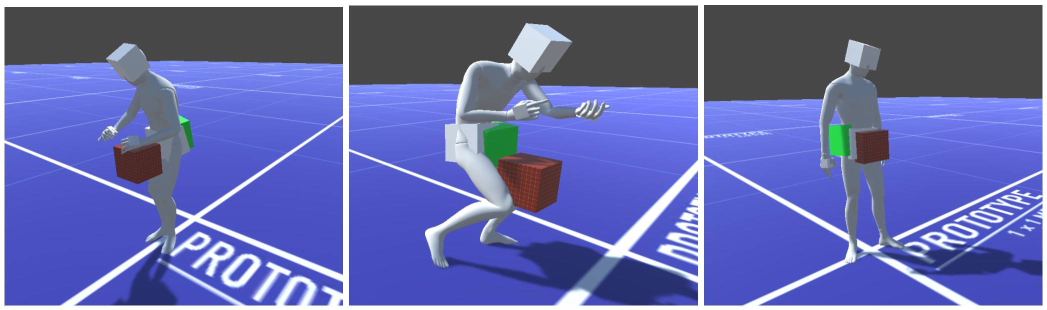

Figure 1: Predictions of proposed model (green) and naïve model (red) compared to the ground-truth waist prediction (white). The proposed model is a substantial improvement over the naïve estimate, and is likely usable in a production scenario.

Figure 1 shows 3D visualizations of the model results on some of the test motions. The proposed model results are shown in green. A naïve model that uses a fixed vertical offset from the head is shown for comparison. Our neural architecture delivers a substantial improvement over the naïve model. The results are quite close to the ground-truth, and are likely usable as-is on a production device.

Figure 2 shows a failure mode of the model (a fast throwing motion). For particularly violent motions, the waist often acquires an unrealistic upward tilt. These motions may be particularly difficult for the network, as they require accurate interpolation over short timescales. Adding velocity inputs into the network could help resolve this issue; we did not have time to investigate this modification.

4 Conclusion

We have demonstrated a practical neural architecture for the prediction of waist position and orientation using only tracking information for the head and hand controllers. We have successfully reproduced the results from [1], and demonstrate improvements in tracking accuracy using training data augmentation and modifications to the original architecture. Our results could allow for a more immersive VR gameplay experience without the need for additional waist tracking hardware.

References

[1] Adityan Jothi, Powen Yao, Andrew Zhao, Mark Miller, Sloan Swieso, and Michael Zyda. Toward predicting user waist location from vr headset and controllers through machine learning. In *Symposium on Spatial User Interaction*, SUI '21, New York, NY, USA, 2021. Association for Computing Machinery.

[2] Juyoung Lee, Andreas Pastor, Jae-In Hwang, and Gerard Jounghyun Kim. Predicting the torso direction from hmd movements for walk-in-place navigation through deep learning. In *25th ACM Symposium on Virtual Reality Software and Technology*, VRST '19, New York, NY, USA, 2019. Association for Computing Machinery.

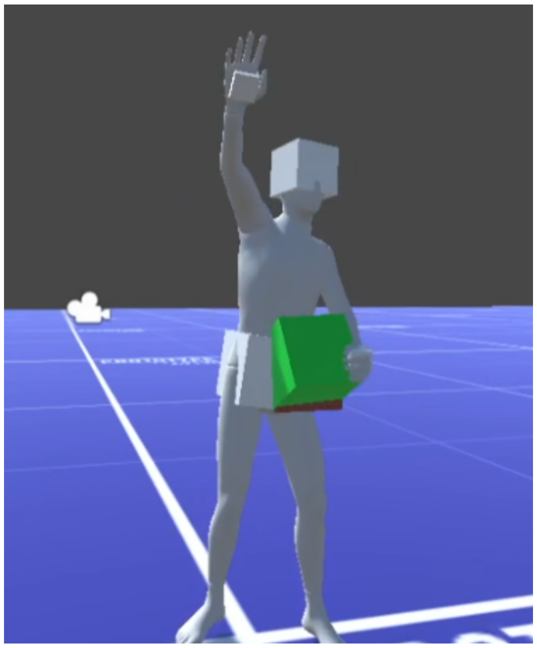

Figure 2: The model often gives inaccurate results for very rapid motions. Here, the model prediction exhibits an unrealistic upward tilt for a violent throwing motion.

Contributions

- Cheuk Ho Yun: Generation of dataset and visualization of results (Unity3D).
- Samuel Li: Torch code for data augmentation/training/evaluation. Reproduction of model and results from [1].
- Uzair Mirza: Model modifications.

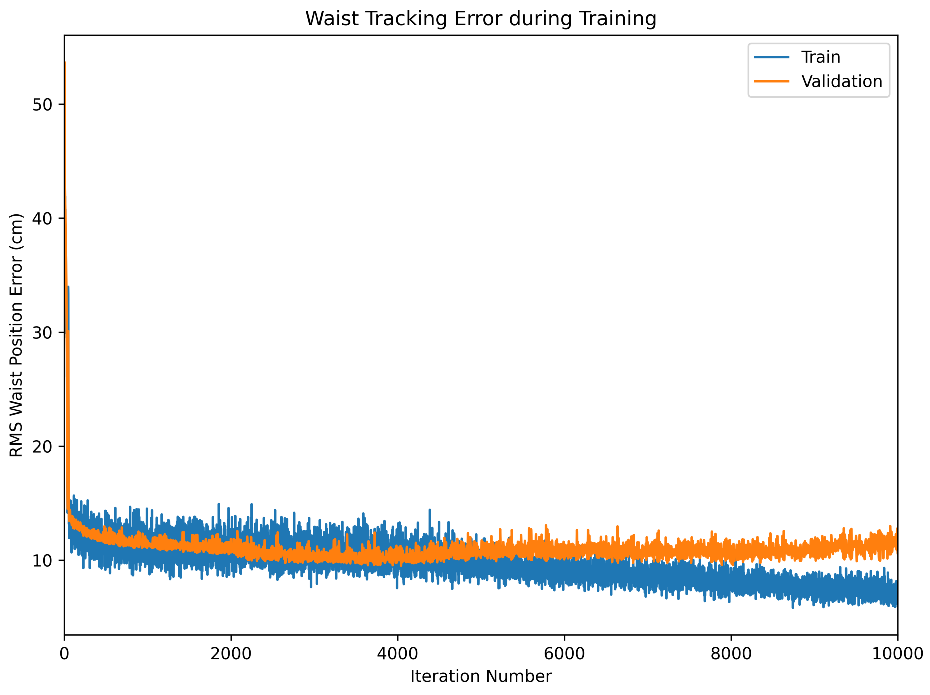

Figure 3: RMS waist positioning error for training (blue) and validation (orange) data during training. Model is a reproduction of [1].

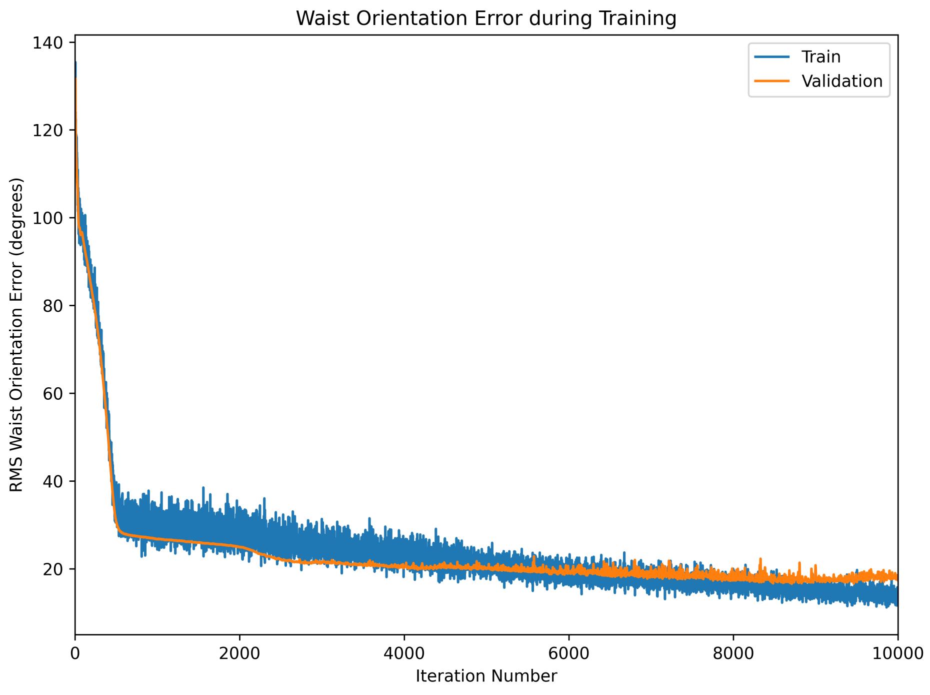

Figure 4: RMS waist rotation error for training (blue) and validation (orange) data during training. Values are approximated using the dot product loss, so early values are an overestimate.

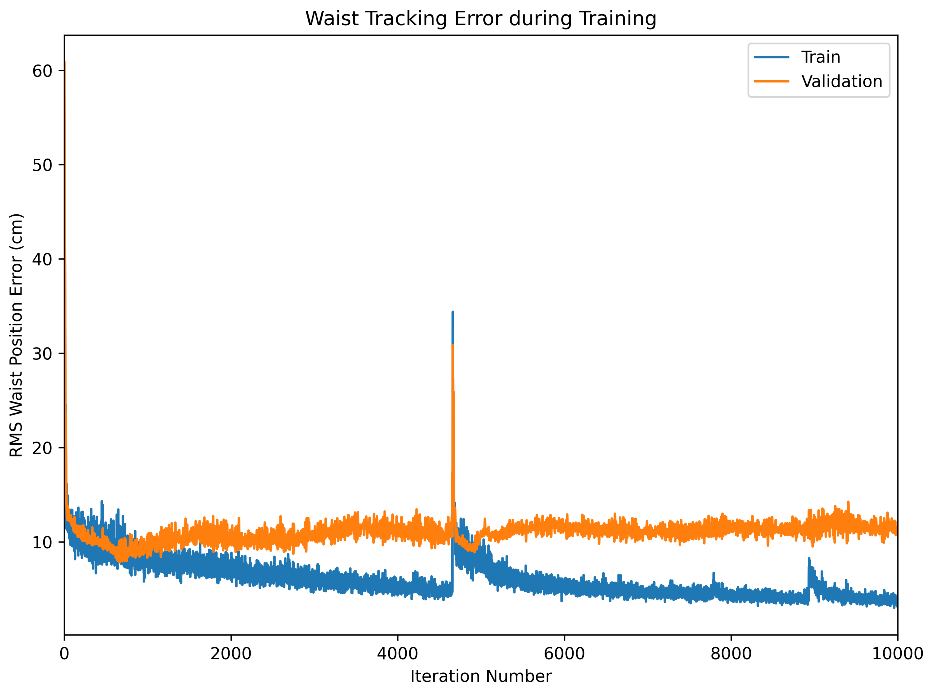

Figure 5: RMS waist positioning error for training (blue) and validation (orange) data during training. Model is a reproduction of [1]. Data augmentation has been disabled.

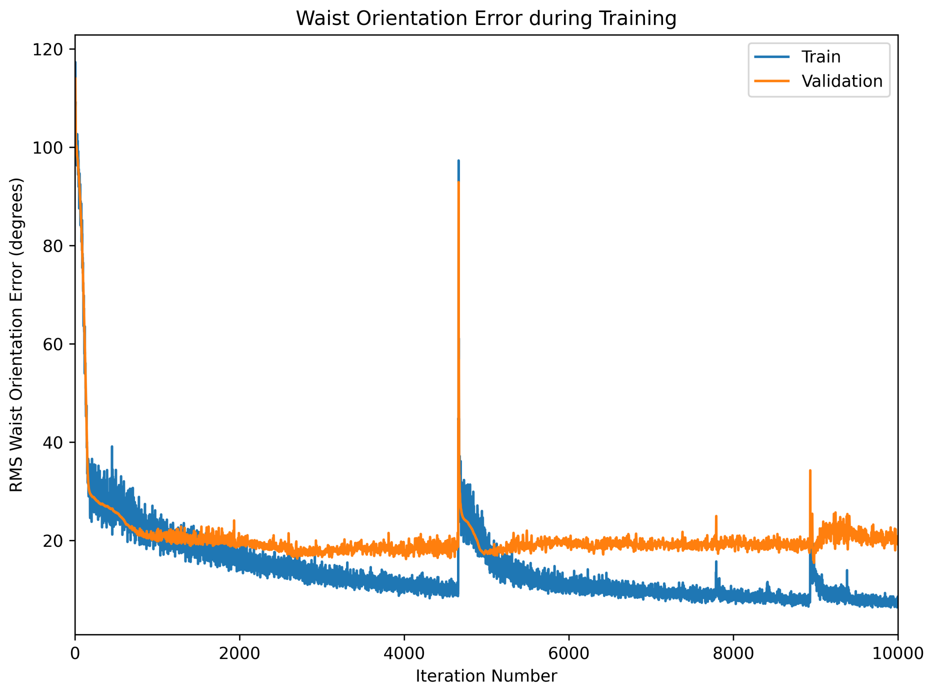

Figure 6: RMS waist rotation error for training (blue) and validation (orange) data during training, using the architecture in [1]. Data augmentation has been disabled.