Sensor Fusion for Learning-based Tracking of Controller Movement in Virtual Reality

1st Chen Song
Computer Science and Engineering
University at Buffalo, the State University of New York
Buffalo, New York
csong5@buffalo.edu

2nd Shuayb Zarar
Microsoft Research
Redmond, Washington
shuayb@microsoft.com

***Abstract*—Inside-out pose tracking of hand-held controllers is an important problem in virtual reality devices. Current state-of-the-art combines a constellation of light-emitting diodes on controllers with a stereo pair of cameras on the head-mounted display (HMD) to track pose. These vision-based systems are unable to track controllers when they move out of the camera's field-of-view (out-of-FOV). To overcome this limitation, we employ sensor fusion and a learning-based model. Specifically, we employ ultrasound sensors on the HMD and controllers to obtain ranging information. We combine this information with predictions from an auto-regressive forecasting model that is built with a recurrent neural network. The combination is achieved *via* a Kalman filter across different positional states (including out-of-FOV). With the proposed approach, we demonstrate near-isotropic accuracy levels (~ 1.23 cm error) in estimating controller position, which was not possible to achieve before with camera-alone tracking.**

***Index Terms*—Ultrasound Ranging, Virtual Reality, Pose Estimation, Autoregression, Recurrent Neural Networks**

I. Introduction

Virtual reality (VR) systems provide users with a remarkable interactive computer-generated experience. For these systems, an ability to render the correct controller position is vital to increasing user immersion. Two approaches have been used to determine the position of hand-held controllers in real-time as they move around the user: inside-out tracking [1], [2] and outside-in tracking [3], [4]. In the former, cameras and sensors are located directly on the head-mounted display (HMD) and controllers, while in the latter, they are placed at a stationary location [5]. Although inside-out tracking systems compromise on tracking accuracy, they help reduce costs and improve deployment efficiency. They are also the focus of this work. A primary method used for inside-out tracking is based on stereo vision, where a pair of cameras track a group of visible or infrared light-emitting diodes (LEDs) as they move around in 3D space. Researchers have demonstrated accurate tracking of these LEDs with decimeter-level errors [6]. However, these techniques do not work well in the presence of heavy sunlight or when the controllers (and the LEDs attached to them) move out-of-FOV. These limitations restrict tracking to a conical volume in front of the cameras, and hurt the interactive user experience [7].

Existing methods to improve tracking performance in VR have primarily focused on the HMD. Authors in [8] have proposed to compute the position and orientation of the HMD *via* magnetic-field analysis. Other approaches have used mounted cameras to capture LED beacons and infer head motion [9]. One of the first papers to look into tracking controller pose is [10], where the authors have utilized controller information for gesture recognition. There are also several commercial VR systems available that employ inside-out methods for controller tracking [11]–[13]. In this paper, we consider a Windows mixed reality (MXR) headset from Samsung as a baseline [12]. We instrument this HMD and the associated controllers with ultrasound transceivers, which we utilize for real-time distance estimation based on time-of-flight (ToF) calculations. We combine this information with predictions of position from a recurrent-neural network (RNN) model to accurately track the controller pose in 3D space. Next, we present details of our system.

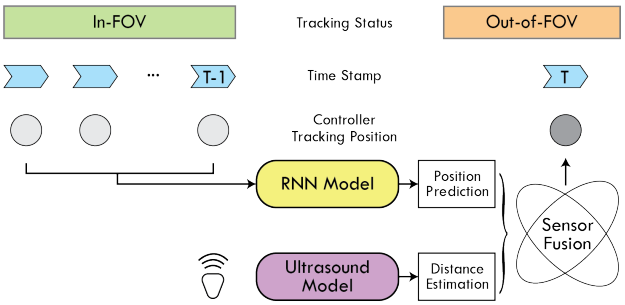


Fig. 1. The proposed approach fuses distance and position estimates from an ultrasound ranging model and an RNN model, respectively for out-of-FOV tracking.

II. System Approach

An overview of our methodology is shown in Fig. 1. It comprises: (1) an ultrasound system to determine a spatial sphere where the controller is likely located, (2) a RNN prediction model to learn continuity in the controller movement: utilizing the past controller position information to forecast the future, and (3) a fusion system that integrates the aforementioned ranging and prediction data to generate the final prediction of the controller position. In this section, we provide details of each of these subcomponents.

978-9-0827-9703-9/19/$31.00 ©2019 IEEE

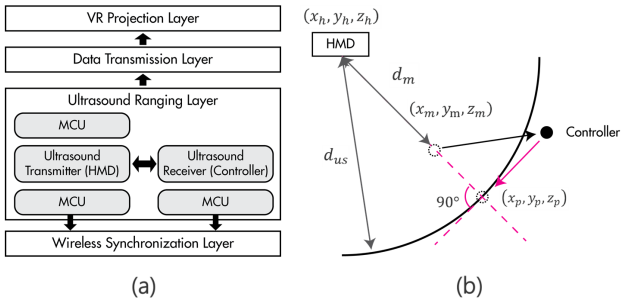


Fig. 2. Proposed ultrasound system: (a) has separate synchronization and data layers, (b) enables 1D-localization.

A. Ultrasound-based Distance Estimation

First, we estimate the distance between controllers and the HMD. The key idea of ultrasound distance estimation is based on ToF analysis. Suppose, we place an ultrasound transmitter on the HMD and a receiver each on the controllers. The distance $d(t)$ at any time t between a controller and the HMD can be estimated as follows:

$$\begin{aligned} d_{us}(t) &= \nu_{us} * t_{tof} \\ &= (331 + 0.6 * T) * t_{tof}, \end{aligned} \tag{1}$$

where, ν_{us} is the speed of sound (a function of temperature T) and t_{tof} is the ToF of the ultrasound pulse.

Time synchronization. Since the ultrasound transmitter and receivers are independently operated, they need to be synchronized in time in order to compute distance $d_{us}(t)$ based on Eq. (1). This is a non-trivial process because the transmitter and receiver operate wirelessly and are driven by separate clocks. In our case, the synchronization process needed to be computationally inexpensive because of energy limitations of the micro-controller (MCU) and real-time constraints of the system. Thus, we utilized a separate light-weight WiFi-based synchronization layer shown in Fig. 2(a). This and the communication layer were implemented based on the user datagram protocol (UDP) that were operating on non-interfering channels [14]. Within the synchronization layer, we employed a statistical method to model delay distributions and compensate for environmental disturbances and UDP protocol overheads. Limited MCU capabilities also affected the design complexity of ultrasound sensing. Through multiple transmitter-receiver pairs, it is possible to achieve 3D localization. However, this incurs extra computation, which was not sustainable on the MCUs that we used. Thus, we limited ourselves to a single transmitter and receiver that were able to produce distance estimates $d_{us}(t)$ in near real-time.

B. Learning-based Trajectory Prediction

Suppose, $\vec{c}(t) = (x_c, y_c, z_c)$ and $\vec{h}(t) = (x_h, y_h, z_h)$ represent the true 3D controller and HMD positions, respectively in the room-coordinate system. Note that $\vec{h}(t)$ is always provided by the VR platform. We utilized in-FOV position estimates $[\vec{c}(1), \ldots, \vec{c}(k)]$, which were also available from the VR platform, to train a model that could predict future positions $\vec{m}(t)$, for all $t \in (k+1, \ldots, k+p)$. This constituted a forecasting problem, which could be solved through well-known autoregressive-modeling techniques like ARMA, ARIMA *etc* [15]. After some experimentation, we decided to use a long short-term memory (LSTM) network because it gave the best predictive performance on sequences of time varying vectors. Specifically, the LSTM network is a deep and recurrent model of neural networks. It takes a stack of features extracted from the input $[\vec{c}(1), \ldots, \vec{c}(k)]$, and produces a refined stack of feature maps entering in the learned prior module. The LSTM works by sequentially updating an internal state, according to the values of three sigmoid gates. Typically, the update is driven by the following equations:

$$\begin{aligned} I_t &= \sigma(W_i \times \vec{c}_t + U_i \times H_{t-1} + b_i) \\ F_t &= \sigma(W_f \times \vec{c}_t + U_f \times H_{t-1} + b_f) \\ O_t &= \sigma(W_o \times \vec{c}_t + U_o \times H_{t-1} + b_o) \\ G_t &= tanh(W_c \times \vec{c}_t + U_c \times H_{t-1} + b_c) \\ C_t &= F_t \odot C_{t-1} + I_t \odot G_t \\ H_t &= O_t \odot tanh(C_t), \end{aligned} \tag{2}$$

where the I_t, F_t and O_t are the 3D activation vectors for input gate, forget gate as well as output gate. H_t is the hidden state vector which stores the memory and C_t is the output cell state vector. The usage of these so-called memory blocks instead of conventional hidden cells which allow them to access and model a self-learned amount of long-range temporal context [16], [17]. Its ability to learn from the past makes LSTM a good candidate to predict continuous movements of the controller [18], [19].

C. Data Fusion for Pose Tracking

It is important to note that in the absence of any additional information, continuous predictions $\vec{m}(t)$ from the machine-learning model are reliable only up a small number of future samples (typically $p \approx 1 - 10$, depending on the prediction frequency). Thus, if the controller goes out-of-FOV for a long duration of time, the LSTM model alone is not able to track position accurately. To overcome this limitation, we periodically correct position estimates using the ultrasound system described in the previous section. Thus, we rectify deviations in the LSTM model (based on low-rate ultrasound measurements) and are able to keep predictions accurate for longer duration of time.

Consider the scenario where a user moves the controller from the cameras view to out-of-FOV (*e.g.*, drawing a sword from the back or holding a shield on the side in a VR game). As mentioned before, once the controller goes out-of-FOV, camera-based systems lose track of it and our goal is to properly estimate the controller position. We do so by fusing 3D-position predictions from the trained LSTM model with distance estimations from the ultrasound sensor. We utilize distance estimates $d_{us}(t)$ from the ultrasound ranging system to localize the controller position in one-dimension as shown in Fig. 2(b) [on to the surface of the sphere $(x - x_h)^2 + (y - y_h)^2 + (z - z_h)^2 = d_{us}^2$]. Specifically, for

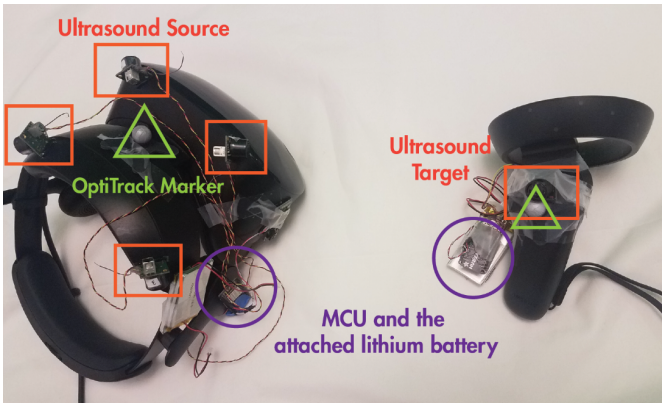


Fig. 3. Ultrasound ranging system integrated with the HMD.

each prediction $\vec{m}(t) = (x_m, y_m, z_m)$ from the LSTM model, we compute vector distances from the HMD as follows:

$$d_m(t) = ||\vec{m}(t) - \vec{h}(t)||_2$$

We fuse distance output from the model $d_m(t)$ with $d_{us}(t)$, obtained from the ultrasound ranging system (separate observation channel), using a Kalman filter (KF), which is a popular choice for estimating user motion in VR applications [20], [21]. We combine the KF distance output, denoted by d_f, with knowledge of the sphere that is centered around the HMD on which the controller is likely located. This is achieved by finding the minimum projection of $\vec{m}(t)$ on the sphere, which is basically a point that lies along the line that passes through $\vec{m}(t)$ and $\vec{h}(t)$, *i.e.*, the center of the sphere. Thus, the minimum projection [final position estimate $\vec{p}(t) = (x_p, y_p, z_p)$] is obtained *via* the following relationship:

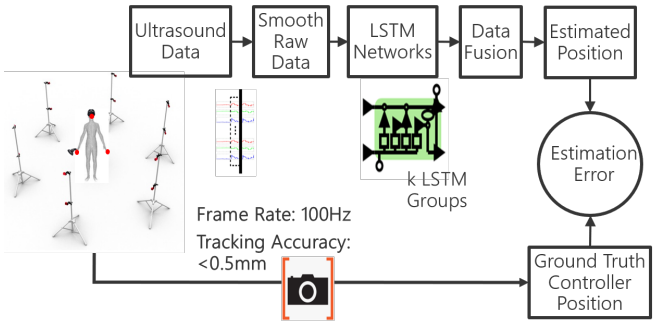
$$\vec{p}(t) = \frac{d_{us}(t)}{d_f(t)} \times \left[\vec{m}(t) - \vec{h}(t)\right] + \vec{h}(t).$$

III. Experimental Results

In this section, we present details of our evaluation system and studies on ultrasound ranging and tracking performance.

A. Hardware and Software Setup

As shown in Fig. 3, we attached four ultrasound transmitters to the HMD in order to cover the 360-degree space. We used one ARM Cortex M0 based MCU to sequentially transmit initialization beacons and ultrasound pulses (unmodulated 42 kHz) from the four transmitters. Each controller also had the same MCU, which managed an ultrasound receiver that was operated in a continuous scan mode. The two MCUs were synchronized over WiFi every 50 ms, and they utilized 200 ms to send-and-receive data cyclically through the four ultrasound transceivers. We used another 50 ms window to compute ultrasound distance and upload the data to a desktop PC *via* a different WiFi UDP channel. With careful mean filtering of the delays, we were able to achieve time synchronization of under 230 us (deviation of 272 us). We also used sensor power levels that allowed us to transmit ultrasound pulses and



Hardware	
MCU (Cortex M0)	80 MHz, 4 MB Flash, 3.3 V, 478 mW
Ultrasound sensor	42 kHz, 20-100 cm (range), 3.3 V
Software	
MXR toolkit	SteamVR + Unity
Base desktop	Python, C#
Ground-truth	8 Flex3 Camera + Motive 2.0.2

Dataset	
Number of trials	10
Avg. length of trial	12 mins
Ultrasound ranging rate.	4 Hz
OptiTrack sampling freq.	100 Hz
MXR sampling freq.	50-60 Hz

Fig. 4. Evaluation framework and detailed specifications.

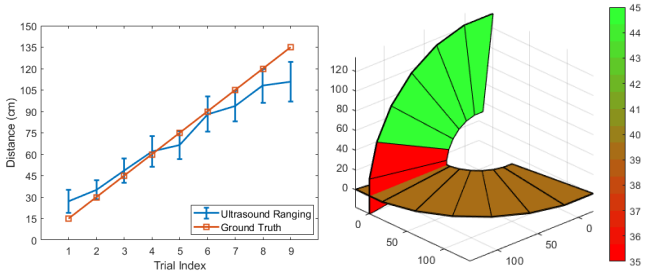
estimate distances in the range 20-100 cm [22]. The MCUs and sensors were powered by a portable lithium (500 mAh) battery. For simplicity, we only present experimental results for tracking a single controller.

Our software system and modeling framework were built in Python and C#. We utilized SteamVR SDK within the Unity environment to collect tracking data from the intrinsic camera-based system [23]. Specifically, we obtained HMD position $\vec{h}(t)$, controller position $\vec{c}(t)$, and the tracking status (*i.e.*, whether the controller was inside-the-FOV or out-of-FOV). The data was sampled at 50 Hz. We developed our LSTM models in TensorFlow. For real-time operation, we implemented the proposed prediction models locally on a PC with 16 GB DDR4 RAM, 3.4 GHz, 16 core 2x Intel Xeon CPUs, and an Nvidia GTX 1080p GPU. We trained multiple LSTM models on a GPU cluster and converged on using 2 recurrent layers with 1024 units. We used the Adam optimization algorithm, mean-squared error loss function, and a dropout rate of 0.2 between the layers.

B. Ground Truth and User Data Collection

In order to obtained ground-truth for the controller position, we used the OptiTrack motion-capture system with eight Flex3 cameras [24]. Through periodic re-calibration, we ensured that the ground-truth had sub-millimeter accuracy. The sampling rate of the OptiTrack system was 100 Hz, and we used the latest version of Motive software to process the data [25]. After an internal approval process, we invited subjects in the age group of 26-36 for data collection. Each of them performed trials based on the required motion with the MXR system instrumented with the custom ultrasound rig. To avoid movement fatigue, we limited data-collection to 10 trials. Each trial lasted for 12 minutes and the subject repetitively performed the motion. The trial was further segmented into a series of complete motion cycles based on the timestamp references

captured by the OptiTrack system. After cleaning and filtering the data, we were able to obtain good measurements for about 2 hours of motion data. Thus, we collected nearly half-a-million position samples in total that were calibrated well with ground-truth data. We used 80% of these samples to train LSTM models, and the rest for evaluation. An overview of the end-to-end experimental setup and details of the data-collection process are summarized in Fig. 4.



(a) We evaluate the ranging performance of different linear distances between the transmitter and the receiver. Generally, the system achieves the optimal performance in the range of 30-60 cm.

(b) We fix the linear distance as 45 cm and evaluate the ranging performance when the transmitter and the receiver have different angles in two axes. The green color indicates high accuracy.

Fig. 5. Performance of ultrasound ranging along the azimuthal axis and altitude.

C. Performance of Ultrasound Ranging

We evaluated the ultrasound ranging system at different 3D-positions of the controller and HMD. First, we varied distances from 15-100 cm, along the azimuthal axis. Fig. 5(a) shows that the ranging system achieves an average tracking error of 9.74 cm. The primary sources of this error were time-synchronization and pulse latching at the receiver. However, in the distance range of 30-60 cm (which is the typical operating distance), the errors were much lower. Error increase with distance could potentially be mitigated by increasing transmission power-levels. We also validated the ranging performance along the altitude (results in the 0-90° range at a fixed linear distance of 45 cm are shown in Fig. 5).

We analyzed ranging accuracies during a complete movement of *drawing-a-sword from the back*; the controller moved from inside-the-FOV to out-of-FOV. As shown in Fig. 6, the MXR system is quite accurate inside-the-FOV. However, it loses track of the controller when it moves out-of-FOV (see the dotted box in the figure). Fortunately, the ultrasound ranging system is still able to track the controller with an average error of 5.82 cm. From the figure, we also observe an increase in the ranging accuracy when the ultrasound estimates are fused with the LSTM prediction *via* the KF.

D. Accuracy of Pose Tracking

Our LSTM-based autoregressive model exploits continuity of hand movements to make accurate predictions of future pose. Fig. 7 shows results from our LSTM model that utilizes 50 prior samples along a sliding window to predict future samples. We see from the figure that the LSTM-model is

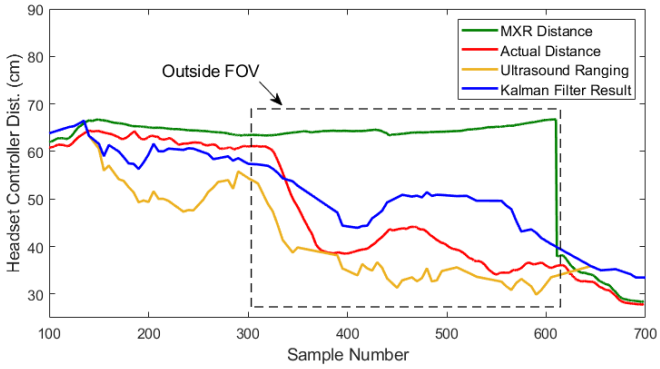


Fig. 6. The results show that the ultrasound ranging is accurate for out-of-FOV tracking. Fusing the ultrasound ranging result with LSTM predictions can boost the system accuracy.

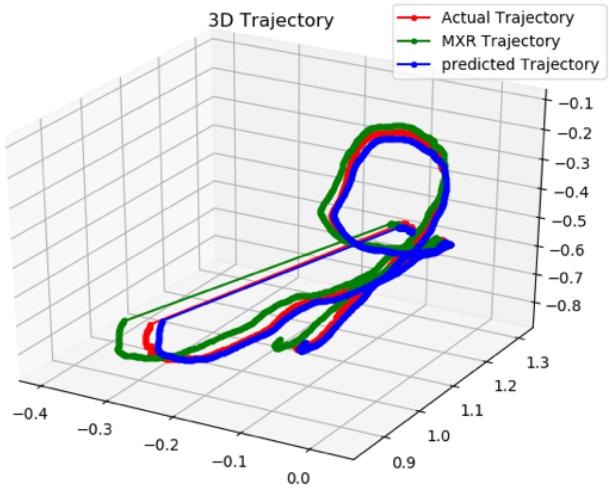


Fig. 7. Predictions from the LSTM model closely track ground-truth pose trajectory. Both axes contain the values in units of meters.

able to closely track the OptiTrack position estimates with an average tracking accuracy of 1.71 cm, which was computed by calculating the Euclidean distance between the prediction sequence and ground-truth position values.

When the controller goes out-of-FOV for large durations of time, LSTM-based autoregression entails re-use of predicted samples for future prediction. This accumulates error in every predicted sample and later pose estimations tend to deviate substantially from the ground truth. To overcome this limitation, we invoke our data-fusion framework (Sec. II-C) that includes rectification through distance measurements obtained from the auxiliary ultrasound system. This periodic correction thus helps bring LSTM-prediction errors back to under 2 cm, when predicting 10 s of samples into the future. An example trajectory is shown in Fig. 8. Future work will involve characterizing the full sample-duration of predictive tracking, tackling issues of non-smooth prediction, improving the neural network architecture with multiple-input-multiple-output design, larger-capacity units, and a combination of several direct h-step estimation models.

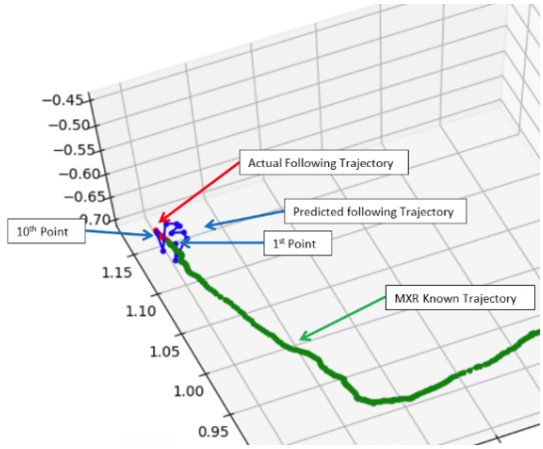


Fig. 8. Our proposed data-fusion system is able to track the controller position over multiple samples into the future. Both axes contain the values in units of meters.

IV. Discussion

Two controllers scenario: For the purpose of demonstration, we reported above how our system can track the position of a single controller. However, two controller scenario is also applicable. Note that the ToF is independently calculated by the ultrasound receiver mounted on each controller based on the broadcast synchronization signal. Once each controller measured its current distance towards to the HMD, it sent the data back to the MCU located on the HMD with its own ID so that later on, the data can be properly differentiated.

Multiple subjects scenario: So far, we only considered one subject scenario because most of the VR games require the physical presence of one player. To support the multiple subjects scenario, the synchronization layers of the transmitter-receiver ranging systems need to be independent from each other so that the synchronization beacons will not be interfered. Specifically, the UDP protocol can be replaced by the transmission control protocol (TCP). However, the propagation and reflection of multiple ultrasound waves in a close environment need to be further considered and studied.

V. Conclusions

In this paper, we developed a methodology to estimate VR-controller positions in real-time when the controllers move out-of-FOV of cameras on the HMD. Our approach was based on fusing distance measurements obtained from an ultrasound-measurement system with position estimations from an LSTM-based autoregression model. Through a complete system and user-collected data, we demonstrated accurate tracking of controllers both in-the-FOV and out-of-FOV with estimation errors of under 2 cm. Thus, our approach enables 360-degree controller tracking, which increases the immersion ability of emerging VR devices.

References

[1] K. Dorfmüller, "Robust Tracking for Augmented Reality Using Retroreflective Markers," *Computers and Graphics*, vol. 23, no. 6, pp. 795–800, 1999.
[2] C. Anthes, R. J. García-Hernández, M. Wiedemann, and D. Kranzlmüller, "State of the Art of Virtual Reality Technology," in *Aerospace Conf.* IEEE, 2016, pp. 1–19.
[3] K. Dorfmüller and H. Wirth, "Real-time Hand and Head Tracking for Virtual Environments Using Infrared Beacons," in *Modelling and Motion Capture Techniques for Virtual Environments*. Springer, 1998, pp. 113–127.
[4] K. Dorfmüller, "An Optical Tracking System for VR/AR Applications," in *Virtual Environments 99*. Springer, 1999, pp. 33–42.
[5] T. Hilfert and M. König, "Low-cost Virtual Reality Environment for Engineering and Construction," *Visualization in Engineering*, vol. 4, no. 1, p. 2, 2016.
[6] H. Kato and M. Billinghurst, "Marker Tracking and HMD Calibration for A Video-based Augmented Reality Conferencing System," in *Proc. Int. Wksp. Augmented Reality*. IEEE, 1999, pp. 85–94.
[7] R. Azuma, "Tracking Requirements for Augmented Reality," *ACM Communications*, vol. 36, no. 7, pp. 50–51, 1993.
[8] F. H. Raab, E. B. Blood, T. O. Steiner, and H. R. Jones, "Magnetic Position and Orientation Tracking System," *IEEE Trans. Aerospace and Electronic Systems*, no. 5, pp. 709–718, 1979.
[9] A. M. Cook, "The Helmet-mounted Visual System in Flight Simulation," *NASA Technical Report*, pp. 1–14, Sep. 1989.
[10] T. Schlömer, B. Poppinga, N. Henze, and S. Boll, "Gesture Recognition with A Wii Controller," in *Proc. Int. Conf. Tangible and Embedded Interaction*. ACM, 2008, pp. 11–14.
[11] "Microsoft Hololens," https://www.microsoft.com/en-us/hololens.
[12] "Samsung Odyssey," https://www.samsung.com.
[13] "Lenovo Daydreamer," https://www.lenovo.com/us/en/daydreamvr/.
[14] "Adafruit Feather HUZZAH ESP8266," https://www.adafruit.com/product/2821.
[15] D. v. Dijk, T. Teräsvirta, and P. H. Franses, "Smooth Transition Autoregressive Models A Survey of Recent Developments," *Econometric Reviews*, vol. 21, no. 1, pp. 1–47, 2002.
[16] S. Hochreiter and J. Schmidhuber, "Long Short-term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[17] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to Forget: Continual Prediction with LSTM," in *Proc. Int. Conf. Artificial Neural Networks*. IET, Sep. 1999, pp. 10–16.
[18] J. Liu, A. Shahroudy, D. Xu, and G. Wang, "Spatio-temporal lstm with trust gates for 3d human action recognition," in *European Conference on Computer Vision*. Springer, 2016, pp. 816–833.
[19] J. Martinez, M. J. Black, and J. Romero, "On human motion prediction using recurrent neural networks," in *Int. Conf. Computer Vision and Pattern Recognition*. IEEE, Jul. 2017, pp. 4674–4683.
[20] E. Foxlin, "Inertial Head-tracker Sensor Fusion by A Complementary Separate-bias Kalman Filter," in *Proc. Int. Symp. Virtual Reality*. IEEE, 1996, pp. 185–194.
[21] J. L. Marins, X. Yun, E. R. Bachmann, R. B. McGhee, and M. J. Zyda, "An Extended Kalman Filter for Quaternion-based Orientation Estimation Using MARG Sensors," in *Int. Conf. Intelligent Robots and Systems*, vol. 4. IEEE, 2001, pp. 2003–2011.
[22] "MaxSonar MB1330," https://www.microsoft.com/en-us/hololens.
[23] "SteamVR SDK," https://www.microsoft.com/en-us/hololens.
[24] "OptiTrack Flex System," https://optitrack.com/products/flex-3/.
[25] "OptiTrack Motive," https://optitrack.com/products/motive/.