RF-Pointer: A Novel Approach to Radar-Driven Pointer Technology for HCI

Elvis Dsouza1, Kevin Chetty2, Shelly Vishwakarma1
Department of Electronic and Computer Science, University of Southampton, UK
† Department of Security and Crime Science, University College London, UK
eld1n22@soton.ac.uk, k.chetty@ucl.ac.uk, s.vishwakarma@soton.ac.uk

***Abstract*—This paper introduces the "RF-Pointer," an innovative radio-frequency-based interaction framework designed for real-time tracking and projection of hand movements in virtual spaces. The system can provide a hands-free and distinctive interaction experience suitable for emerging domains like virtual reality and augmented reality. In this study, we delve into the architectural and operational dynamics of the RF-Pointer, utilizing a prototype equipped with a 77 GHz radar sensor. Initial tests reveal an average error of 2.5 cm in estimating the pointer's location. To further illustrate the efficacy of the proposed architecture, we conduct a qualitative comparison with ground truth pointer location data, presenting the results in the form of tracked trajectories corresponding to both ground truth and the RF-Pointer estimated trajectories. The tracking results demonstrate that RF-Pointer trajectories closely align with ground truth trajectories. Consequently, these findings underscore the potential for improved accuracy through expanded data training, positioning the RF-Pointer as a significant advancement with the capability to transform interactive technology.**

***Index Terms*—Radar, Continuous Gesture Recognition, Human-Computer Interaction, Millimeter-Wave Radar, Privacy-Preserving HCI, Virtual Reality.**

I. Introduction

Radio-Frequency (RF) sensing is transforming gesture recognition in Human-Computer Interaction (HCI), providing users with a natural and intuitive means of engaging with technology through nuanced movements [1]. Unlike traditional methods requiring direct interaction or visual cues, RF sensors excel in capturing and interpreting fine-grained human gestures from a distance, eliminating the need for direct contact or a clear line of sight. This not only enhances user privacy and convenience but also expands accessibility, making technology interaction feasible in diverse settings where other sensory systems like infrared, ultrasonic, or camera may prove impractical. The integration of cost-effective millimeter-wave RF sensors, offering high-resolution and accurate detection at an affordable price, is further driving this technological advancement. Combined with sophisticated artificial intelligence (AI) algorithms, average gesture recognition accuracies exceeding 90% have been achieved. However, a notable limitation in existing works is the focus on recognizing predefined gestures, constraining users to a fixed set of motions [2]. This rigidity may hinder user-friendly interactions, especially in emerging domains like virtual reality (VR) and augmented reality (AR) [3].

In the VR/AR landscape, achieving a fully immersive experience relies on precisely tracking hand movements. Human hands exhibit a diverse range of expressive motions, from subtle gestures to spontaneous movements, which predefined gestures may not fully capture. While significant strides have been made in enhancing hand-tracking technologies using various sensors, there exists a noticeable research gap when it comes to leveraging radio-frequency sensors for this purpose.

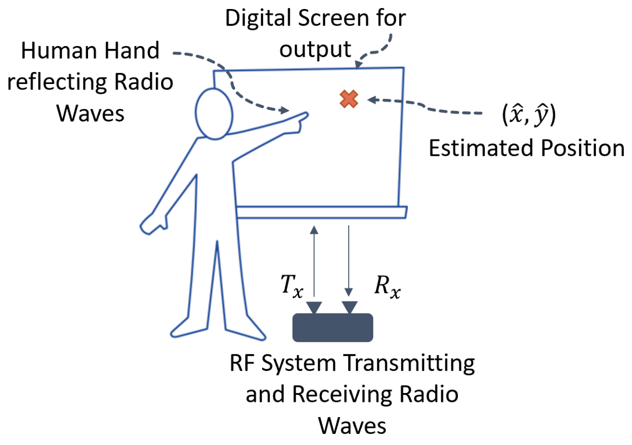

Human Hand reflecting Radio Waves
Digital Screen for output
$(\hat{x}, \hat{y})$
Estimated Position
T_x
R_x
RF System Transmitting and Receiving Radio Waves

Fig. 1: RF Pointer Overview

To address this gap, we present a novel RF-based pointer system, RF-Pointer, designed to enhance navigation in the VR domain. Drawing inspiration from radar-based gesture recognition systems like Project Soli [4] and ThuMouse [5], RF-Pointer allows users to explore virtual environments naturally and intuitively, contributing to a more immersive VR experience. The system operates by emitting and detecting radio waves that interact with a human pointing hand, translating these movements into on-screen actions as clearly shown in Fig.1. Time-of-flight measurements determine distances, while Doppler shift analysis reveals hand velocity. The system employs sophisticated signal processing to generate a three-dimensional point cloud representing hand position in space, which can be mapped onto the digital or virtual screen.

In developing the RF-Pointer, we employ a millimeter-wave (mmWave) radar sensor known for its compact design and low power consumption. This sensor's characteristics make it exceptionally well-suited for seamless integration into VR headsets. RF-Pointer generates a three-dimensional point cloud representing hand position in space. However, inherent inaccuracies in the gathered point cloud data due to environmental conditions and reflections necessitate refinement. To address

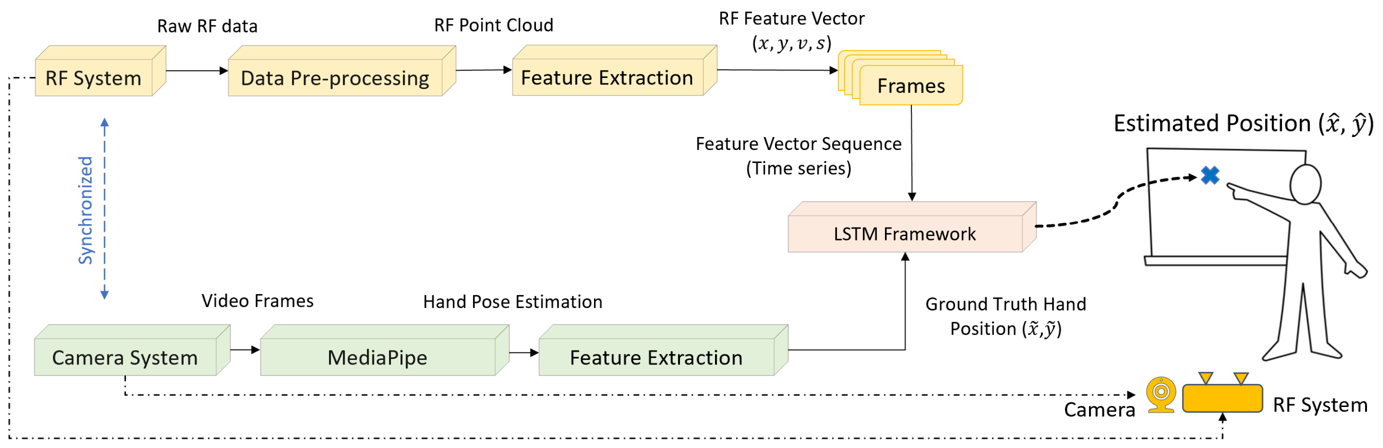


Fig. 2: RF-Pointer System Framework

this, we employ a Long Short Term Memory (LSTM) network, trained on a paired dataset of ground truth point cloud data and radar point cloud data, to correct temporal dynamics and spatial coordinates. This iterative refinement maximizes the reliability and precision of RF-Pointer. We gather ground truth point cloud data from a camera sensor running a pre-trained hand tracking model from Google's MediaPipe [6].

To assess the performance of the RF-Pointer, we conduct a comprehensive evaluation, involving both qualitative and quantitative comparisons between our radar-based pointer location estimation and ground truth data. The resulting trajectories of hand tracking from the radar closely align with those obtained from the ground truth data, providing strong evidence of the effectiveness of our approach.

Our paper is structured as follows: In Section II, we present an in-depth discussion of the proposed RF-Pointer framework. This section encompasses details on the chosen interaction paradigm, the experimental setup, and the data pre-processing procedure involving two synchronized systems – a measurement RF system and a camera system. Following this, Section III outlines the data collection pipeline, presenting associated results and analyses. We conduct a thorough comparison of the RF-Pointer's performance with the ground truth data, both qualitatively and quantitatively. Finally, in Section IV, we conclude our paper with insights and considerations for future improvements to the system.

II. RF-Pointer Framework

Fig. 2 provides an overview of the proposed RF-Pointer system architecture. The system consists of a digital screen displaying the pointer location, similar to a laser pointer, and two synchronized sensors: an RF sensor and a camera sensor. For the RF sensor, we employ Texas Instrument's (TI) AWR1642BOOST[7], and for the camera sensor, we use the laptop's built-in web camera. This section outlines the process of collecting point cloud data from the radar and corresponding ground truth data from a camera system running MediaPipe for extracting hand landmarks. Additionally, we detail how we train a Long Short-Term Memory (LSTM) network with these training pairs, specifically for estimating the location of the pointing human hand and removing any outliers from the radar data. Each component is further discussed in detail in the following sections.

A. Interaction Paradigm

We start by making foundational assumptions: the virtual space, where interaction outcomes are presented, is a two-dimensional surface, such as a computer screen or a projection wall, while the actions or gestures occur in the three-dimensional real-world space in front of this surface. In this study, our emphasis is on two-dimensional virtual surfaces, like monitor screens. To maintain simplicity, we adopt a natural pointing hand pose characterized by extending the index finger while keeping the other fingers in a nearly clenched fist position—an intuitive and widely applicable choice for pointer applications.

Fig. 3 illustrates the three primary interaction methods conducive to pointing systems for interacting with virtual applications on a 2D surface. The first is relative motion tracking, commonly found in computer peripherals, which simplifies design but lacks spatial precision for natural human motion. This method translates controller movements into virtual pointer shifts, making it unsuitable for precise hand-pointing in AR/VR systems. The second is surface projection, which produces a 2D representation of the controller at the tracked pointer's position, enabling 3D movement but neglects screen distance. The third is 3D Extrapolation, as depicted in Fig. 3(c). 3D Extrapolation integrates position and orientation data for precise pointing in three dimensions by projecting the controller's angles onto the virtual surface, facilitating manipulation across vast distances with minimal effort.

In this study, we utilize TI's AWR1642 radar, capable of gathering only 2D point cloud data instead of 3D point clouds. Consequently, we choose the 2D Projection approach from the available pointer design methods, as it offers the most intuitive interface.

B. RF System and Associated Data Pre-processing

We use TI's AWR1642BOOST radar which is a high bandwidth (4GHz) FMCW (Frequency Modulated Continuous

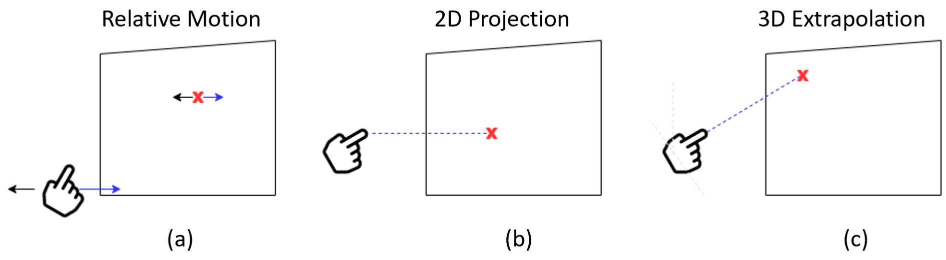


Fig. 3: Interaction Paradigm-(a) Relative motion with hand movement and virtual pointer movement, (b) 2D Projection of hand to the screen and (c) 3D extrapolation of pointing direction to the screen

Wave) radar with operational frequency range between 76-81GHz. The radar has four receiving and two transmitting antennas, enabling tracking multiple objects. Notably, unlike other TI radars, this radar features receivers arranged in a 1D antenna array, allowing detection along either the azimuth angle or the elevation angle [8]. We present radar's operating parameters in Table I.

Parameter	Value
Azimuth Resolution	15°
Range Resolution	0.039 m
Maximum unambiguous Range	11.99 m
Maximum Radial Velocity	1.41 m/s
Radial velocity resolution	0.18 m/s
Range Detection Threshold	15 dB
Doppler Detection Threshold	15 dB

TABLE I: Radar Configuration

Data Pre-processing: We transmit a sequence of linearly frequency-modulated sinusoid waveforms, commonly known as chirps, from the radar. As these chirps encounter a moving hand, they undergo backscattering, resulting in complex signals that represent time-varying reflections—a combination of attenuation, time delay, and Doppler shift in comparison to the transmitted signal. The delay in the reflected signal corresponds to the range of the pointing hand, while the phase difference between consecutive chirps indicates the relative velocity between the radar and the hand. The complex reflectivity is proportional to the radar cross-section. The raw time domain signal undergoes preprocessing through a 2-dimensional Fourier Transform to extract range (r) and velocity information (v). Furthermore, we perform Fast Fourier Transform (FFT) along the receivers to acquire azimuth angle information (ϕ). Simultaneously, we capture velocity profiles and reflected signal strength (s) in the form of Signal-to-Noise Ratio (SNR) for each scatter. The challenge arises from the varying number of scatterers detected in each frame, prompting the selection of the scatter with the highest normalized SNR value in each frame. This ensures that each frame includes the position, relative SNR, and velocity of the most prominent scatterer with the highest reflected signal strength.

Fig. 5 illustrates our experimental setup, positioned on the table directly below a human hand pointing towards a digital screen. The detection plane is parallel to the screen. On the detection plane, the movement of the hand along the vertical axis is extracted from the range information (r), and the movement along the horizontal axis is gathered from the azimuth information (ϕ). This process is clearly depicted in Fig. 4.

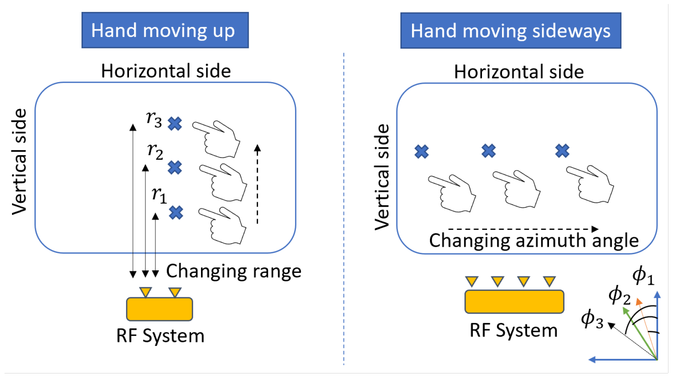


Fig. 4: Interaction plane in the RF field

We convert the detected hand's position from radial coordinates (r, ϕ) to Cartesian coordinates (x, y) using the equations $x = r \times \cos(\phi)$ and $y = r \times \sin(\phi)$. This transformation yields 2D point cloud data (x, y), including the velocity (v) and SNR (s) of each detected point in a 2D Cartesian plane per frame. To reduce unnecessary noise from a broad radar

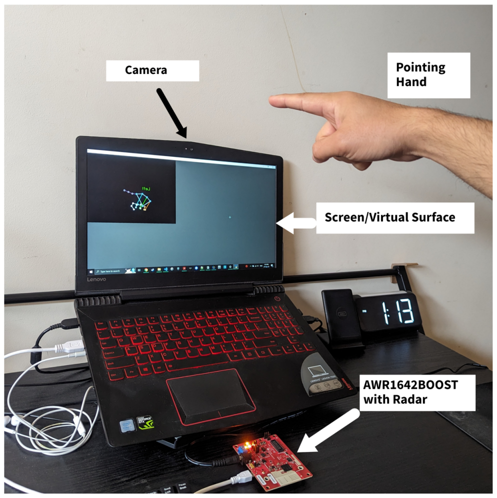


Fig. 5: Experimental Setup comprising of TI AWR1642BOOST radar sensor, web-camera and a human subject pointing towards the monitor.

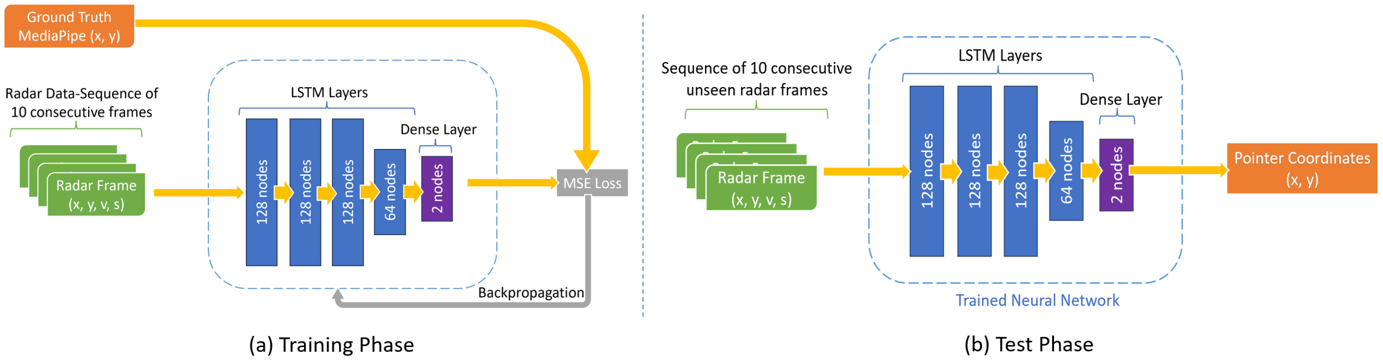


Fig. 6: Deep Learning Architecture, (a) Training phase with simulatneous inut feature vectors from radar (x,y,v,s) and MediaPpe output $(\tilde{x}, \tilde{y})$, (b) Test phase estimating pointer position only using radar data

sensing field, we narrow the effective interaction field to align with the screen's size. Empirically chosen dimensions are employed to ensure that the system captures only intentional interactions occurring right in front of the screen, filtering out most unintentional hand motions.

C. Ground Truth Camera System and MediaPipe

We employ Google's Mediapipe Hand Landmark Detection pipeline, utilizing a webcam to estimate the human hand pose and assess the accuracy of the resulting pointer location . This pipeline utilizes a pre-trained machine learning model within Google's Mediapipe library, specifically designed for hand landmark estimation. This model employs a deep neural network architecture to analyze input data in the form of a camera image or monochrome video, predicting the spatial locations of key landmarks on the hand $(\tilde{x}, \tilde{y})$, including fingertips, knuckles, and the palm.

The Mediapipe computer vision library offers estimates for 21 hand landmarks, covering various points from the wrist to individual finger joints. To optimize for pointing applications, we concentrate solely on the normalized position (0, 1) of the index finger within the camera image frame, excluding additional pose data.

D. Deep Learning

We collect output from the radar in the form of 2D point cloud data (x, y), along with the velocity (v) and SNR (s) for each frame, and simultaneously obtain the tracked location $(\tilde{x}, \tilde{y})$ using a camera system through MediaPipe. Ideally, the 2D point cloud location detected by the radar should always match the one detected by the ground truth camera system, implying $\tilde{x} = x$ and $\tilde{y} = y$. However, this is not always the case due to various factors affecting radar-based tracking of hands. These factors include the interaction of dynamic radar scatter centers from different parts of the moving hand, multipath interactions, occlusion, environmental noise, and instances where the SNR is too low.

Therefore, we design an LSTM network that effectively learns to map the temporal dynamics of radar data to the spatial coordinates of hands tracked by MediaPipe while correcting for any outlier scattering centers [9]. Fig.6(a) illustrates the training phase of our proposed deep learning architecture, comprising five layers. The first layer consists of 128 LSTM nodes, the fourth layer has 64 LSTM nodes, and the final output layer has two densely connected nodes mapped to normalized $\tilde{x}$ and $\tilde{y}$ positions of the ground truth.

During the training process, the LSTM receives a sequence of 10 radar frames as input, with each frame represented as a vector of 4 values: $[x, y, v, s]$, where x and y are the normalized positions of the object from radar, v is the detected velocity of that object, and s is the normalized SNR value. Simultaneously, it is provided with ground truth data from MediaPipe for each time step, encompassing true position information. To synchronize radar and camera data with varying frame rates, we adjust the timestamped vision frames to match the constant frame rate of the radar system.

The LSTM excels at learning temporal patterns inherent in sequential data, capturing dependencies and relationships between radar measurements and ground truth. As it processes the input data, the LSTM generates predictions for the tracked scatterer's state, incorporating estimates of position and velocity. Subsequently, the network refines its predictions through a comparison with ground truth data, adjusting internal parameters during training to minimize disparities. This iterative training involves backpropagation through time (BPTT) to optimize the LSTM's predictive accuracy.

Once trained, during the test phase (shown in Fig. 6b), the LSTM demonstrates proficiency in generalizing to unseen data, effectively serving as a tracking filter for estimating a scatterer's state based solely on the radar data.

III. Experimental Implementation

A simple experiment is conducted to assess the system's viability using the outlined methodology and design.

A. Data Collection Pipeline

To capture the pointing hand motion, the radar and vision pipelines ran simultaneously in four sessions, each lasting three minutes and collecting data at a consistent frame rate of 30 fps. Processing of the data was carried out on a Lenovo Laptop equipped with an Intel Core i7-7700HQ CPU, 32 GB RAM, and Nvidia GTX 1050 GPU. The unique position of

Parameter	Value
Error in $\hat{x}$	2.8063 cm
Error in $\hat{y}$	2.2165 cm

TABLE II: Model Training and Loss

the single response object inside the ROI was extracted after pre-processing, and the data was organized into sequences, each comprising 10 frames. In total, 21,271 sequences, each containing 10 frames, were generated for training the neural network model.

The radar receiver array was strategically positioned to ensure optimal vertical buffer space from the camera, capturing gestures within an interaction space rectangle. The resulting 2D point cloud data, representing detected objects with x-y coordinates, velocity, and Signal-to-Noise ratio (SNR), underwent processing in Python. To ensure synchronized radar and vision streams, a correction for a 70-frame lag at the start of the radar sensor was implemented during alignment in each session. Following alignment, spatial filtering, and data centering, a region of interest (ROI) of 40cm x 30cm was defined for interactions, with objects outside this ROI being disregarded.

B. Network Training Parameter Settings

We empirically found the following parameter settings to be the most effective- optimization with adaptive moment estimation (ADAM) with an initial learning rate of of 0.001, gradient decay factor of 0.9 and squared gradient decay factor of 0.99. The learning rate is updated for every 200 epochs with a batch size fixed to 32. We utilise TensorFlow on Google Colaboratory with a Tesla T4 GPU to train our LSTM network and employed mean squared error loss function for effective model training.

C. Experimental Results and Analysis

We use 80% of the synchronized radar measurements and camera measurements for training the LSTM network, reserving the remaining 20% for testing the model's performance in estimating the pointer location on the screen. The accuracies of the resulting pointer location estimations are presented in Table II. The outcomes reveal an error of 2.8 cm along the horizontal axis and 2.21 cm along the vertical direction in the pointer's location estimation.

In regards to processing time, our model has a size of 2.31 MB on disk, comprising 380,290 model parameters. Signal processing tasks were delegated to the microcontroller unit on the radar evaluation module, capturing data at a frame rate of 30 fps. Table III illustrates the total computational time involved in the pre-processing and data preparation pipeline, which is less than 0.4838 ms.

1) ***RF-Pointer Visualisations in 2D Space****:* To rigorously assess the performance of the RF-Pointer, we conducted additional experiments, visualizing the estimated pointer location against the expected output gathered using MediaPipe. Following the protocol specified in the previous section, we present the comparison for a number of pointer trajectories.

Task	Time duration (ms)
Digital Signal Processing (DSP)	$<$ 33.333
Data processing	0.4837
Model Prediction	0.0001457
Total (excluding DSP)	0.4838457

TABLE III: Latency per component

It's important to note that in all figures, the blue solid line represents the ground truth data, and the yellow cross represents the pointer location predicted by the RF-Pointer framework.

Figure 7 displays the experimental results for six different sessions, each lasting 3 seconds. In the first two sessions, the user performed two specific gestures: one involved drawing a vertical line through the pointing finger, and the second gesture required drawing a tick sign in front of the screen. The four other gestures included random motions, which were simultaneously captured by the camera module for ground truth. Across all sessions, we observe that the estimated RF pointer location closely follows the dynamic path of the ground truth trajectory for most of the time. However, there are also some outliers, which could be eliminated using sophisticated signal processing algorithms such as simple thresholding or a focused Random Sample Consensus (RANSAC). As this is a preliminary study, the implementation and comparison of solutions for outliers are beyond the scope, and we leave this aspect for subsequent works.

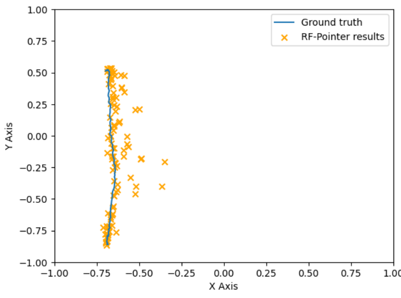
2) ***Performance Benchmarking****:* We performed a comparative analysis of the proposed RF-Pointer and existing pose detection methods such as RFPose3D and WiPose, presenting the resulting estimation accuracies in Table IV. The results indicate that RF-Pointer demonstrates superior hand tracking accuracy. This improvement can be attributed to the fact that RF-Pointer exclusively focuses on tracking hands, whereas other methods concurrently track both hands and the entire body pose. Given that our primary emphasis is on pointing applications, such as VR interactions and other Human-Computer Interaction (HCI) interactions that specifically involve hand movements, the emphasis on hand tracking is justified. This focused approach not only yields good accuracy but also provides room for further enhancements.

Model	Average Error (mm)	Hand Joints Error (mm)
WiPose	28.3	97.5
RFPose3D	36.7	150.5
RF-Pointer	NA	**25.1**

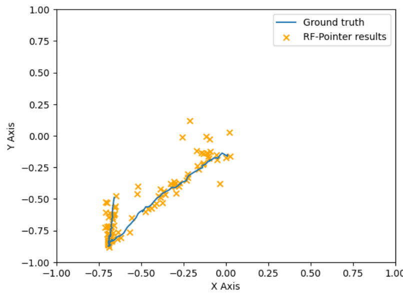
TABLE IV: Performance benchmarking against WiPose and RFPose3D for Joint Localisation Error

IV. Conclusions and Future Work on Addressing Challenges

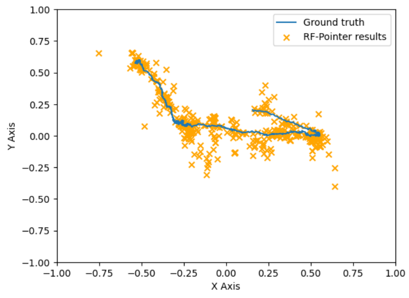
We demonstrate the efficacy of the RF-Pointer architecture by achieving an average error rate of 2.5 cm in estimating the pointer location on a digital screen. The RF-Pointer system not only competes with leading technologies but also surpasses pose estimation solutions. In contrast to methods reliant on pre-defined gestures, our approach is trained on a dataset



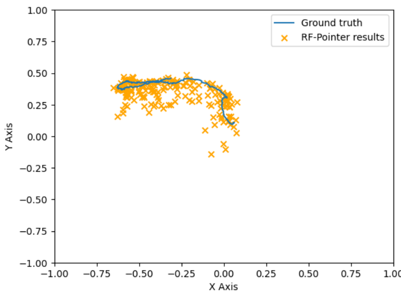
(a)



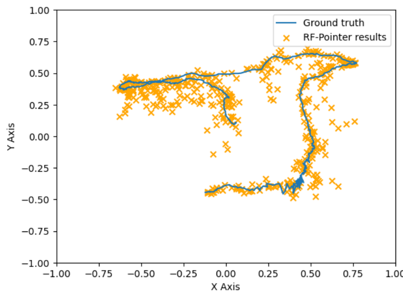
(b)



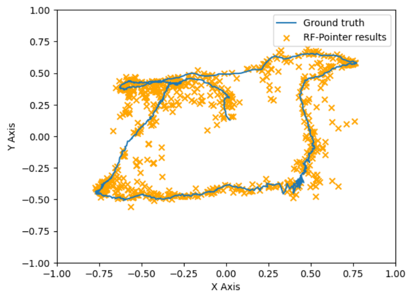
(c)



(d)



(e)



(f)

Fig. 7: Qualitative results from a human performing different gestures with a pointing finger, (a) drawing a vertical line, (b) drawing a tick sign, (c)-(f) some random trajectories. Note that pointer location is estimated based on only radar data. Ground truth hand land-marker data is extracted using a camera module running MediaPipe Model.

comprising radar data with point cloud information, velocity, and SNR features from various continuous gestures. Ground truth data, essential for training, is obtained using an in-built web camera running a pre-trained deep learning model from Google's MediaPipe Hand tracking library. The RF-Pointer effectively mitigates environmental effects from the estimated pointer location, enabling accurate tracking of human hands across random gestures. Qualitative results further confirm the accuracy, aligning radar-tracked trajectories with ground truth data for diverse pointing gestures. The RF-Pointer shows promise for seamless integration into VR and other Human-Computer Interaction (HCI) systems, facilitating hands-free interaction with screens. Nevertheless, we acknowledge the limitations of our study and propose potential improvements for future research.

1) Synchronization between the radar and camera sensors was achieved through empirical observations and corrections for data capture delays. However, addressing synchronization challenges is crucial. In future work, we will explore hardware triggering methods such as master-slave configurations, external triggering sources, or software synchronization based on timestamps and a common clock source.
2) Our model was trained with a limited dataset captured from a single human subject, indicating the need for improved performance through training with more diverse data to enhance the generalization capability of the entire framework.
3) Ground truth data was captured using MediaPipe, which may not always provide accurate data. To enhance accuracy, future research will employ a more sophisticated motion capture-based hand tracking system.

References

[1] A. Patra, P. Geuer, A. Munari, and P. Mähönen, "mm-wave radar based gesture recognition: Development and evaluation of a low-power, low-complexity system," in *Proceedings of the 2nd ACM Workshop on Millimeter Wave Networks and Sensing Systems*, 2018, pp. 51–56.
[2] S. Franceschini, M. Ambrosanio, S. Vitale, F. Baselice, A. Gifuni, G. Grassini, and V. Pascazio, "Hand gesture recognition via radar sensors and convolutional neural networks," in *2020 IEEE Radar Conference (RadarConf20)*. IEEE, 2020, pp. 1–5.
[3] P. Wang, S. Zhang, X. Bai, M. Billinghurst, W. He, S. Wang, X. Zhang, J. Du, and Y. Chen, "Head pointer or eye gaze: Which helps more in mr remote collaboration?" in *2019 IEEE conference on virtual reality and 3D user interfaces (VR)*. IEEE, 2019, pp. 1219–1220.
[4] S. Wang, J. Song, J. Lien, I. Poupyrev, and O. Hilliges, "Interacting with soli: Exploring fine-grained dynamic gesture recognition in the radio-frequency spectrum," in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, 2016, pp. 851–860.
[5] Z. Li, Z. Lei, A. Yan, E. Solovey, and K. Pahlavan, "Thumouse: A micro-gesture cursor input through mmwave radar-based interaction," in *2020 IEEE International Conference on Consumer Electronics (ICCE)*, 2020, pp. 1–9.
[6] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C. Chang, and M. Grundmann, "Mediapipe hands: On-device real-time hand tracking," *CoRR*, vol. abs/2006.10214, 2020. [Online]. Available: https://arxiv.org/abs/2006.10214
[7] T. Instruments, "Awr1642boost evaluation board," 2023. [Online]. Available: https://www.ti.com/tool/AWR1642BOOST
[8] ——, "Awr1642 single chip radar," 2023. [Online]. Available: https://www.ti.com/product/AWR1642
[9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, p. 1735–1780, nov 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735