An FCNN-Based Super-Resolution mmWave Radar Framework for Contactless Musical Instrument Interface

Josiah W. Smith[1], Orges Furxhi[2], and Murat Torlak[1]

[1]Dept. of Electrical and Computer Engineering, The University of Texas at Dallas, Richardson, TX, United States
[2]Camera Systems and Computational Imaging, Imec, Kissimmee, FL, United States

***Abstract*—In this paper, we propose a framework for contactless human-computer interaction (HCI) using novel tracking techniques based on deep learning-based super-resolution and tracking algorithms. Our system offers unprecedented high-resolution tracking of hand position and motion characteristics by leveraging spatial and temporal features embedded in the reflected radar waveform. Rather than classifying sample from a predefined a set of hand gestures, as common in existing work on deep learning with mmWave radar, our proposed imager employs a regressive full convolutional neural network (FCNN) approach to achieve spatial super-resolution improving localization. While the proposed techniques are suitable for a host of tracking applications, this article focuses on their application as a musical interface to demonstrate the robustness of the gesture sensing pipeline and deep learning signal processing chain. The user can control the instrument by varying the position and velocity of their hand above the vertically-facing sensor. By employing a commercially-available multiple-input-multiple-output (MIMO) radar rather than a traditional optical sensor, our framework demonstrates the efficacy of the mmWave sensing modality for fine motion tracking and offers an elegant solution to a host of HCI tasks. Additionally, we provide a freely-available software package and user interface for controlling the device, streaming the data to MATLAB in real-time, and increasing accessibility to the signal processing and device interface functionality utilized in this article.**

***Index Terms*—deep learning, human-computer interaction (HCI), fully-convolutional neural network (FCNN), millimeter-wave (mmWave), multiple-input multiple-output (MIMO), radar perception, super-resolution**

I. Introduction

Radar perception for human-computer interaction (HCI) on multiple-input-multiple-output (MIMO) millimeter-wave (mmWave) radars has emerged as a promising solution to a variety of sensing problems. The physical nature of millimeter-waves offers a safe method for high-resolution imaging where optical sensors may fail due to insufficient lighting, fog, or other line-of-sight interference. Additionally, mmWave sensors are considered less-invasive than optical counterparts and promote user privacy. Ultra-wideband MIMO devices enable centimeter-level spatial resolution with a small profile device. As a result, precise spatial information of a target scene can be easily acquired from such imaging devices at a low cost.

mmWave sensors are relatively modern technology, but some of the earliest electronic interfaces were contactless devices for physically expressive musical control including the Radio Drum and Theremin [1]. Russian physicist Leon Theremin demonstrated his noncontact musical instrument in 1921, an interface controlled by the proximity of the musician's hand to an antenna using beat-frequency oscillators and a capacitive sensing apparatus [2]. More recently, computer vision approaches have been adopted for the innovation of contactless new musical interfaces (NMIs), most of which rely on optical camera solutions. Extensive prior work exists on optical-based NMIs using popular sensors such as the Microsoft Kinect and Leap Motion.

In [3], Polfreman uses the Kinect to track the 3-D position of both hands of a standing performer to construct a multi-modal instrument. Trail *et al.* present a pitched percussion hyper-instrument to track the tips of two mallets simultaneously with the Kinect [4]. Crossole, designed by Senturk *et al.*, is a Kinect-based metainstrument that visualizes chord progressions as virtual blocks resembling a crossword puzzle [5]. Schramm *et al.* use the Kinect to analyze and classify motions of a orchestral conductor [6]. In [7], the Kinect is used to track hand motion across time and then translated to music using the inverse Fourier transform of the physical pattern using a sonification technique called sonomotiongram.

Alternatively, the popular Leap Motion controller is capable of modeling the entire hand, including the fingers, which allows for even more detailed hand posture-based gesture control to be explored for musical interface development. Using the Leap Motion sensor, Han *et al.* developed two NMIs, *Air Keys* and *Air Pad*. *Air Keys* tracks the motion and position of each finger to recognize when and which keys the musician is pressing and playing the desired notes. Similarly, *Air Pad* tracks the hand position to create a 2-D virtual drum pad played by pressing specific regions in a 2-D horizontal plane, thus requiring accurate 3-D hand-tracking [8]. Hantrakul and Kaczmarek use the Leap Motion controller to track both hands for controlling MIDI (Musical Instrument Digital Interface) instruments and virtual effects [9]. Similarly, Leimu pairs the Leap Motion with an inertial measurement unit (IMU) demonstrating improved performance over the Leap Motion controller alone for musical interface [10]. Other solutions have been attempted, such as employing non-invasive force sensing resistors to enhance "traditional" instruments by learning and monitoring for gestures performed by the musician [11].

In the optical HCI domain, [12] proposes a musical interface using only a portable RGB camera to recognize hand gestures using a gesture classification technique. Akbari and Cheng developed a system to transcribe music played on a piano in real-time using optical cameras positioned to view the keys [13]. These projects have yielded high-performing real-time musical interfaces capable of consistent high-accuracy motion tracking but require several key design constraints, namely specific lighting conditions and line-of-sight. As shown in this article, mmWave sensors overcome these major obstacles while providing superior privacy through the means of advanced spatiotemporal algorithms. However, little work has been done towards gestural musical interfaces on mmWave radar sensors using hand-tracking techniques. Even though extensive research exists on static and dynamic gesture recognition using deep learning models and mmWave radars [14], [15], Google ATAP's Project Soli is the only effort using mmWave radar for musical interface, using gesture recognition and 1-D position estimation to control the parameters of audio synthesizers [16].

The novel framework presented in this article offers a major advancement for near-field mmWave hand-tracking and an accessible MATLAB software platform for further investigation into real-time mmWave HCI and algorithm innovation. 2-D localization performance is considerably improved from past work [17] by employing a novel deep learning-based technique to improve the resolution beyond the theoretical limitations.

It is important to note our approach is contrary to gesture classification, wherein the objective is to determine the class of a sample from a set of predefined classes as in [14], [15]; rather, we apply a novel fully convolutional neural network (FCNN) to preserve the geometry of the image and perform super-resolution for improved localization. Prior work on resolution improvement using FCNNs has been limited to the far-field domain with large apertures [18]; however, our novel approach unifies FCNN-based super-resolution with near-field imaging on a small (8-channel) array and is shown to improve hand-tracking performance significantly. Additionally, a particle filter tracking algorithm is presented to further improve tracking robustness by employing the Doppler effect. Compared to prior work on gesture tracking using optical solutions [3], [7]–[9], [12], [19], our approach offers fine hand-tracking using a single mmWave sensor offering higher depth resolution with superior privacy. This article proposes a novel hand-tracking method for musical interface by fusing spatiotemporal algorithms, deep learning-enhanced feature extraction, and robust position tracking algorithms. To aid further development and prototyping for real-time mmWave gesture applications, the entire software implementation is available by request to the corresponding author. To our knowledge, this proposed framework is the first openly available software package supporting real-time data streaming from a mmWave radar into MATLAB for streamlined signal processing and deep learning algorithm development.

The rest of this paper is formatted as follows. Section II provides an overview of the frequency modulated continuous wave (FMCW) radar signal model and feature extraction methods. In Section III, two robust tracking algorithms and estimation techniques are presented. The system implementation is discussed in Section IV and results are shown in Section V. Section VI provides a discussion of the performance, design constraints, and distinct advantages of the two tracking methods in Sections III-A and III-B, followed finally by conclusions.

Notation: Throughout this paper, vectors and matrices are set in boldface, using lowercase letters for vectors and uppercase letters for matrices. The superscripts ^{T} and * denote the transpose and conjugation operations, respectively. The identity matrix of size $N \times N$ is expressed as $\boldsymbol{I}_N$ and $\mathbf{1}_N$ is the all-ones vector of size $N \times 1$. Spatial coordinates are treated as continuous to support continuously distributed target scenes and all time variables are modeled in discrete-time.

II. Preliminaries of MIMO-FMCW Radar Signaling

In this section, we overview the propagation model for the FMCW radar chirp signal and examine the spatiotemporal features of a target in motion. The imaging scenario, as shown in Fig. 1, consists of a multistatic linear MIMO array facing vertically. Orthogonality is leveraged in time by employing time-division-multiplexing MIMO (TDM-MIMO), wherein the transmitters are activated at separate time instances. Throughout this paper, the musician's hand is modeled as a point reflector located at the point (y, z), an assumption that holds given the physical limitations of the device and scenario examined in this article and has been verified empirically.

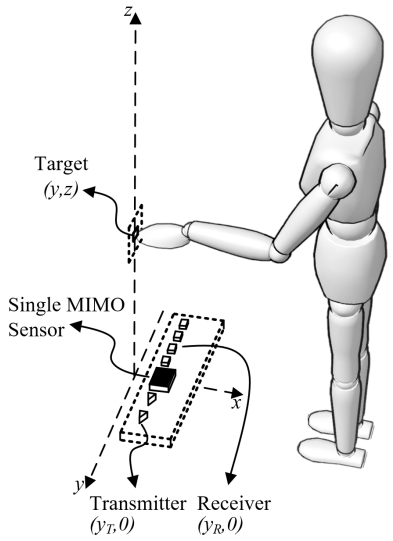

Fig. 1. The imaging geometry, where the linear MIMO array faces vertically and the musician moves their hand throughout the y-z plane.

A. MIMO-FMCW Signal Model

The FMCW chirp signal model is well documented in literature [20]–[22] and is discussed in this section for reference and continuity throughout this paper. Considering a single transmitter/receiver pair located at $(y_T, 0)$ and $(y_R, 0)$ in the y-z plane, respectively, and an ideal point target with reflectivity

p located at (y, z), the time sampled FMCW beat signal can be expressed in discrete-time as

$$s(y_T, y_R, n_k) = \frac{p}{R_T R_R} e^{j(k_0+\Delta n_k)(R_T+R_R)}, \tag{1}$$

where R_T, R_R are the distances from the transmitter and receiver to the point target, respectively, n_k is the wavenumber index, $k_0 = 2\pi f_0/c$ is the starting wavenumber corresponding to the starting frequency f_0, and $\Delta = 2\pi K/(cf_S)$ is the wavenumber step size with K being the chirp slope, f_S being the sampling frequency, and c is the speed of light.

To ease the subsequent signal processing, it is desirable to approximate the multistatic MIMO beat signal, represented in (1) as its corresponding monostatic equivalent using the approximation developed in [23] as

$$\hat{s}(y', n_k) = s(y_T, y_R, n_k) e^{-j(k_0+\Delta n_k)\frac{d_y^2}{4z_0}}, \tag{2}$$

valid only for small d_y, the distance between the transmitter and receiver elements, where z_0 is a reference plane typically given as the center of the target scene. Taking y' as the locations of the virtual elements located at the midpoints between each transceiver pair and R as the corresponding distance from each virtual element to the point reflector, the resulting monostatic beat signal approximates to

$$\hat{s}(y', n_k) \approx \frac{p}{R^2} e^{j2(k_0+\Delta n_k)R}. \tag{3}$$

From (3), the spatial location, (y, z), of the target is embedded in the radar beat signal, in the form of the radial distance R.

B. Doppler Radar Signal Processing

The relative velocity of a target can be extracted from the beat signal expressed in (3) by exploiting the Doppler effect. As discussed in [24], by transmitting a series of chirp waveforms at a known pulse repetition interval (PRI), T_{PRI}, the velocity of a moving target can be identified as the frequency component along the chirp index dimension given by

$$\hat{s}(y', n_k, n_c) = \frac{p}{R^2} e^{j(2(k_0+\Delta n_k)R + \frac{4\pi v T_{PRI}}{\lambda_0} n_c)}, \tag{4}$$

where R is the initial range of the target, v is the velocity of the target, λ_0 is the wavelength corresponding to f_0, and n_c is the chirp index,

Thus, the beat signal sampled across time is a 2-D complex sinusoidal with frequencies corresponding to the range and velocity of the target on the first and second dimensions, respectively. Subsequently, to extract the range and velocity, traditional methods perform a 2-D fast Fourier transform (FFT) over a matrix whose rows or columns consist of subsequent chirps.

C. Range Migration Algorithm Image Reconstruction

To achieve high-fidelity 2-D localization, we employ the range migration algorithm (RMA) over traditional range-angle FFT methods [20], whose localization accuracy is known to be inferior [25]. The primary goal of the RMA is to reconstruct the target scene's reflectivity function, $p(y, z)$. For a distributed target, the beat signal can be modeled as the superposition of the backscattered signal at every point in the scene, neglecting the amplitude terms, as

$$\hat{s}(y', n_k) = \iint p(y, z) e^{j2(k_0+\Delta n_k)R} dy dz, \tag{5}$$

This target model assumes a spatially distributed target whose reflectivity only depends on spatial location and neglects the any frequency dependence of the reflectivity function. Inverting (5) using the method of stationary phase, the reflectivity function, $p(y, z)$, can be estimated efficiently by

$$\hat{p}(y, z) = \text{IFT}_{\text{2D}}^{(k_y, k_z)} \left[\mathcal{S} \left[\text{IFT}_{\text{1D}}^{(y')} [\hat{s}^*(y', n_k)] \right] \right], \tag{6}$$

where $\mathcal{S}[\bullet]$ is the Stolt interpolation operation [23] and FT[$\bullet$], IFT[$\bullet$] are the forward and inverse Fourier transform operators. To avoid aliasing in the image sampling criteria must be considered [22]. Spatial resolution along the y and z directions are constrained by the physical and device limitations and are expressed as

$$\delta_y = \frac{\lambda_c z_0}{2D_y}, \tag{7}$$

$$\delta_z = \frac{c}{2B}, \tag{8}$$

where λ_c is the wavelength corresponding to the frequency at the center of the chirp sweep, D_y is the aperture size along the y direction, and z_0 is the center of the imaging scene [22].

After the 2-D reflectivity function of the target scene is recovered, the hand position is estimated subsequently as

$$\{\hat{y}, \hat{z}\} = \arg\max_{\{y,z\}} \hat{p}(y, z). \tag{9}$$

Further, the aforementioned Doppler principle can be leveraged to extract the velocity of the target by Fourier analysis over successive chirps. To optimally exploit the deep learning framework discussed in Section III-B2 and reduce the required computation complexity, the velocity is extracted after the RMA is performed and hand location is estimated.

As evident in (4), the velocity is decoupled from the wavenumber index and is the scaled frequency component along the chirp index dimension. As a result, the phase term corresponding to the velocity is preserved in the reconstructed image, $\hat{p}(y, z)$. Therefore, the velocity profile can be obtained by performing an FFT across the chirp index, n_c, dimension of the recent images. Rather than performing the FFT across the 3-D array, $\hat{p}(y, z, n_c)$, we perform the FFT over the slice of the image corresponding to the estimated position, $\hat{y}$, yielding the velocity profile along the z-direction, where n_d is the velocity index, as

$$\hat{d}(z, n_d) = \text{FFT}_{\text{1D}}^{(n_c)} \left[\hat{p}(y, z, n_c) \Big|_{y=\hat{y}} \right]. \tag{10}$$

Finally, the velocity can be estimated from (10) using video pulse integration by

$$\hat{v}_d = \arg\max \sqrt{\int |\tilde{d}(z, n_d)|^2 dz}. \tag{11}$$

The velocity computed by this method is referred to as the Doppler velocity. The recovered velocity using this approach is limited by the timing and physical constraints between $[-\frac{\lambda_0}{4T_{PRI}}, \frac{\lambda_0}{4T_{PRI}}]$. Later, the Doppler velocity is employed to improve the tracking performance using the Doppler corroborated particle filter.

III. Spatiotemporal Imaging on mmWave Radar

In this section, we present the methods for our proposed imager capable of high accuracy hand-tracking for HCI. The contribution of this article is the advancement in algorithm performance for 2-D localization by utilizing both the novel super-resolution FCNN and proposed tracking algorithm. While we will investigate the application of such algorithms as an NMI, our mmWave radar-based sensing algorithms can be applied to a host of HCI problems.

It is important to note that this work is not intended to compete with the computational efficiency of embedded HCI solutions and existing musical interfaces. Rather, the main contributions of this article are novel algorithms for super-resolution spatiotemporal hand-tracking and a freely-downloadable platform to increase accessibility and encourage further research in this arena. As such, we will focus primarily on the development of the algorithms and their localization performance. Discussions on performance and implementation issues are considered secondary and are addressed in Sections IV and VI.

A. Classical Spatiotemporal Feature Extraction Techniques

In this section, we introduce the simple approach to spatiotemporal sensing for contactless musical instrument interface. While our system generally tracks the 2-D position and velocity of the user's hand, we have identified three underlying features to achieve fine control of the musical interface: range, cross-range oscillation, and velocity. By the geometry given in Fig. 1, we define the range as the position of the hand along the z-axis, i.e. the vertical displacement between the sensor and the user's hand. Similarly, cross-range is defined as the position of the hand along the y-axis. Subsequently, cross-range oscillation is the rate at which the hand oscillates in the cross-range direction. Velocity is given by the velocity of the hand with respect to the range z-axis. These parameters are selected such that the output musical interface is controlled primarily by the range of the musician's hand and secondarily by the cross-range oscillation and velocity. However, these parameters can be assigned by the user based on preference using the MIDI interface, as discussed later. Throughout the remainder of this article, we will refer to these parameters as features extracted from the radar beat signal.

Under the simple gesture tracking regime, the 2-D location and velocity $(\hat{y}, \hat{z}, \hat{v}_d)$ are extracted from the reconstructed image and buffer of recent images using (9) and (11). In the next section, the three parameters extracted from the raw data are treated as a vector called the noisy measurement vector $\boldsymbol{r}$. In the optimal scenario, the bandwidth, antenna array size, and the signal-to-noise-ratio (SNR) are quite large, tending towards infinity. For the case of an 8 channel automotive mmWave radar and a human hand, the bandwidth is limited (4 GHz), the antenna array size is small ($D_y = 2\lambda_c$), and the reflectivity of the hand is not high compared to the noise level. As a result, simply extracting the maximum from the reconstructed RMA images yields sporadic location and velocity estimates. Even in the ideal case, the spatial resolution of our system along the y and z directions is $\delta_y = 7.5$ cm and $\delta_z = 3.75$ cm, respectively. Several other factors are not taken into account in the classical, direct tracking method including beam-pattern, residual phase errors, and antenna coupling. All these limitations and non-idealities in the imaging scenario degrade the image and result in noisy location and velocity estimates; however, many of these issues analytical forms and cannot be solved directly classical methods. To address these issues, we present a novel data-driven approach employing an FCNN for super-resolution and image enhancement.

B. FCNN-Based Super-Resolution Feature Extraction and Particle Filter Tracking Methods

In this section, we improve upon the simple tracking techniques to overcome noise and foundational non-idealities in the imaging scenario, yielding a much-improved user experience. The concepts demonstrated in this section are applicable for many tracking and high-resolution imaging applications beyond the scope of musical interfaces.

To improve the tracking robustness of the proposed musical interface, we adopt the well-known particle filter [26] and present a novel modification. While traditional methods such as the extended Kalman filter (EKF) employ a motion model, our implementation of the particle filter bypasses the need for a deterministic motion model. The particle filter is selected for this application as other traditional approaches have demonstrated poor tracking performance in our experimentation, yielding either sporadic localization or overly damped, sluggish estimation. Additionally, the particle filter is advantageous as it can track non-linear dynamics and does not require prior knowledge of the motion model or noise parameters for robust localization.

In our modification of the particle filter, the control input is a weighted movement towards the newest measurement. To demonstrate our proposed algorithm, consider the case of simultaneous location estimation along the y and z directions. The new noisy measurement vector, $\boldsymbol{r}$, has two elements, the newest estimates of location, $\hat{y}$ and $\hat{z}$, which are extracted by the methods described in the prior section. Algorithm 1 details the modified particle filter implementation. For 2-D localization, $\boldsymbol{X}_n$ is a matrix of size $N \times 2$, whose rows are the (y, z) coordinates of each particle at time index n, where N is the number of particles, and $\boldsymbol{w}_n$ is the vector of weights corresponding to each particle. The estimates of the 2-D location (also known as the estimated states) form the vector $\boldsymbol{s}_n$ and the multivariate Gaussian distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ is denoted as $G(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Before executing the iterative algorithm, the initial particle states matrix, $\boldsymbol{X}_0$, and initial weights vector, $\boldsymbol{w}_0$, are initialized with as random locations throughout the region of interest (ROI) and uniform weights, respectively.

Algorithm 1: Modified Particle Filter Algorithm

input : $\boldsymbol{r} = [\hat{y}, \hat{z}]^T$
output: $\boldsymbol{s}_n = [\tilde{y}, \tilde{z}]^T$
1 $\boldsymbol{X}_n \leftarrow$ rows of $\boldsymbol{X}_{n-1}$ sampled using weights $\boldsymbol{w}_{n-1}$;
2 $\boldsymbol{X}_n \leftarrow \boldsymbol{X}_n + \mathbf{1}_N \boldsymbol{a}^T(\boldsymbol{r} - \boldsymbol{s}_{n-1}) + \boldsymbol{\psi}$;
3 $\boldsymbol{w}_n \leftarrow e^{-\frac{1}{2}(\boldsymbol{X}_n - \boldsymbol{s}_{n-1})^T \boldsymbol{\Sigma}_w^{-1} (\boldsymbol{X}_n - \boldsymbol{s}_{n-1})}$;
4 $\boldsymbol{s}_n \leftarrow \frac{1}{\mathbf{1}_N^T \boldsymbol{w}_n} \boldsymbol{X}_n^T \boldsymbol{w}_n$;

Proper handling of the key steps, (step 2) resampling of the particle states and (step 3) computing new weights, is essential to effectively implement our novel particle filter algorithm.

The particle resampling process involves moving the particles towards the new measurement by a specified weight. $\boldsymbol{a} = [a_y, a_z]^T$ is a vector whose two elements provide weight to the noisy estimates $\hat{y}$ and $\hat{z}$, respectively. The size of $\boldsymbol{a}$, $\boldsymbol{r}$, and $\boldsymbol{s}_n$ can be varied depending on the number of parameters to be tracked by the particle filter. Hence, the new measurements do not dominate the motion tracking but have a weighted influence on the localization procedure. Fig. 2 demonstrates the resampling process with $a_y = a_z = 0.5$. Note that before computing the new weights, particle diffusion is performed by adding the perturbation term $\boldsymbol{\psi}$. The random vector $\boldsymbol{\psi}$ is Gaussian distributed with zero mean and predefined covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\psi}}$.

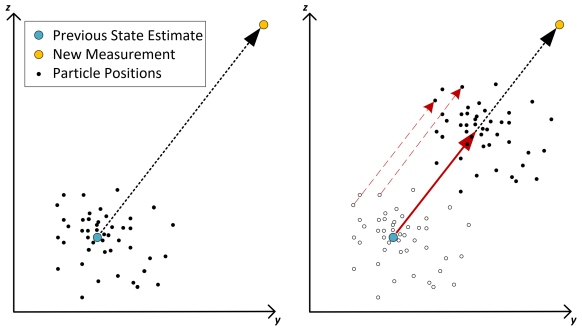

Fig. 2. A visual example of the modified particle filter algorithm resampling process. The particle locations are resampled by a shift transformation towards the new measurement according to the weight vector $\boldsymbol{a}$, where $a_y = a_z = 0.5$.

The new weights are computed from a multivariate Gaussian distribution with the previously estimated states, $\boldsymbol{s}_{n-1}$, as the mean vector and a predefined covariance matrix $\boldsymbol{\Sigma}_w$. Therefore, particles closer to the previously estimated state are assigned a higher weight than those farther away. This results in a tendency towards small changes in the state estimations while monitoring for movement from the current position. For many applications requiring precise and consistent localization and motion tracking on mmWave radar, our modified particle filter algorithm is an ideal fit as it tends to a steady-state estimation of the states but remains active in monitoring the noisy sensor input.

1) Doppler-Corroborated Real-Time Weighting: In this section, we present a dynamic weighting technique for updating $\boldsymbol{a}$ in real-time by exploiting the dependence between position and velocity. Our approach considers corroboration between the Doppler velocity estimate and the velocity estimated from the range samples as a measure of the new measurement's reliability. Thus, the dependability of the Doppler velocity can improve tracking of the target position along the range (z) dimension even in the presence of noisy position estimates. After the Doppler velocity is calculated by (11), the recent range estimates are used to calculate the sample velocity ($\hat{v}_s$) by the least squares estimator as

$$\hat{v}_s = \frac{N_z T_{PRI} \sum_m (\boldsymbol{z}^{(m)} m) - T_{PRI} \sum_m \boldsymbol{z}^{(m)} \sum_m m}{N_z \sum_m (\boldsymbol{z}^{(m)})^2 - \left(\sum_m \boldsymbol{z}^{(m)}\right)^2}, \tag{12}$$

where $\boldsymbol{z}^{(m)}$ is the m^{th} element of the vector of recent $\hat{z}$ estimates, $\boldsymbol{z}$, with $\boldsymbol{z}^{(N_z-1)}$ being the most recent.

The difference between the Doppler estimated velocity and sample estimated velocity is computed as $\Delta_v = |\hat{v}_d - \hat{v}_s|$ and used in the reward function (13) to update the weight placed on the new noisy measurement in real-time.

$$a_z(\Delta_v) = \begin{cases} a_{z,0} \cos\left(\frac{2\pi T_{PRI} \Delta_v}{\lambda_0}\right) & \text{if } \Delta_v \leq \frac{\lambda_0}{4T_{PRI}} \\ 0 & \text{if } \Delta_v > \frac{\lambda_0}{4T_{PRI}} \end{cases} \tag{13}$$

When the sample velocity is close to Doppler velocity, i.e. Δ_v is small, the reward function is close to $a_{z,0}$. Hence, the new measurement is corroborated by the reliable Doppler velocity and weighted accordingly. Outliers and erroneous measurements contradicting the Doppler velocity are given less importance during the particle resampling process. To implement the Doppler corroborated particle filter, $\boldsymbol{a} = [a_y, a_z(\Delta_v)]^T$ is dynamically updated by (13) at each iteration of Algorithm 1.

2) Improved 2-D Position Estimation by Enhancing FCNN: The modified particle filter algorithm improves the tracking consistency and smoothness; however, several issues such as instrumentation delay, ambient/device noise, multistatic effects, and non-spherical beam patterns remain unaddressed and degrade tracking performance. To overcome these non-idealities, we present a novel FCNN-based technique for image enhancement that improves the 2-D position estimation, subsequent tracking accuracy, and Doppler spectrum SNR. Compared to prior FCNN synthetic aperture radar (SAR) techniques employing far-field assumptions and trained on synthetically generated data [18], our enhancement FCNN method operates on near-field images, improves localization even with a small aperture, and is trained using a novel technique allowing the network to learn the environment and device noise, near-field beam pattern, and multistatic effects.

To train the enhancement FCNN, we construct a dataset consisting of both real human hand data and synthetically generated data. Real hand data are collected by capturing frames while the user holds their hand at known locations relative to the device and synthetic data are used to supplement the training set. Each synthetic sample is generated by simulating a MIMO beat signal using (5) with one ideal point target located at a known location and additive real device noise, collected from the radar. The simulated locations are randomized to uniformly cover the ROI. Both the real and synthetic data are

used as features in the FCNN training process, thus enabling the network to fit the non-ideal beam pattern, real multipath and multistatic effects, empirical reflection of a human hand, device and ambient noise, and hand positions throughout the ROI.

To train the image-to-image regression FCNN, each training feature (real or synthetic image) must correspond to a ground truth label. The ground truth label images are synthetically generated by the model

$$\mathcal{I}(y,z) = e^{-(y-y_0)^2/\sigma_y^2 - (z-z_0)^2/\sigma_z^2} \tag{14}$$

where the width of the expected target located at (y_0, z_0) is dictated by σ_y and σ_z in the y and z dimensions, respectively, yielding resolutions of $1.18\sigma_y$ and $1.18\sigma_z$ according to the 3 dB beamwidth definition [27]. Each label is generated using the requisite knowledge of the location of the human hand or target of each feature image. During training, the FCNN learns the highly nonlinear relationship between distorted, blurred RMA images and the ideal images generated using (14). Our novel training technique results in a robust and generalizable FCNN that improves image SNR and localization by fitting to the non-ideal imaging constraints. Further, the trained network enables localization precision beyond the physical limitations of the device improving tracking performance significantly. FCNN training is discussed in Section IV-D and results are presented and discussed in Section V-C.

Additionally, by isolating the peak corresponding to the human hand, clutter and phase noise at other positions are mitigated thereby improving the Doppler spectrum SNR and subsequent velocity estimation. Thus, the FCNN enhances both the spatial and temporal features extracted from the radar beat signal before the particle filter. Uniting the proposed particle filter and enhancement FCNN, the range, cross-range oscillation, and velocity are robustly tracked by our novel algorithms and mapped to musical interface controls.

IV. System Design and Implementation

In this section, we present the system implementation for both the classical tracking techniques and our novel super-resolution feature extraction and tracking algorithms discussed in the previous section.

A. Hardware and Software Implementation

The hardware employed in the proposed system consists of a Texas Instruments (TI) AWR1243 automotive radar in conjunction with a DCA1000EVM real-time data capture adapter. The TI radar is a MIMO-FMCW mmWave radar with an operating bandwidth of 4 GHz and a center frequency of 79 GHz. In this research, we utilize the linear MIMO array consisting of 2 transmit antenna (TX) elements, separated by $2\lambda_c$, and 4 receive antenna (RX) elements, separated by $\lambda_c/2$. The resulting virtual array has 8 equally spaced virtual elements separated by $\lambda_c/4$ [20]. The calibration methodology discussed in [23] is adopted to mitigate range bias, constant phase errors, and instrumentation delay. In this process, data are captured from a corner reflector at a known location and used to identify range bias and phase offsets among the antennas. Unlike an optical or infrared calibration, this process is invariant of lighting and temperature constraints as well as user hand sizes, etc. Thus, the one-time calibration applies to a variety of environments and users.

The software platform for signal processing, visualization, and machine learning is written in MATLAB. Despite its inferior computational efficiency compared to other languages, MATLAB is employed to provide an accessible platform for researchers to engage with this work and rapidly prototype custom real-time algorithms using our custom tools. Once the algorithms are validated on a PC, they can be implemented onto such embedded devices for optimized application-specific usage. For a positive user experience as a musical interface, latency and timing issues must be taken into account, and are discussed in Section VI.

B. Real-Time Data Retrieval and Interactive MATLAB User Interface

To stream the data from the device into MATLAB, a custom UDP interface software is written. This routine is implemented efficiently in C++ and is capable of receiving the sequential UDP packets, organizing the packets to form each chirp, and providing the data to MATLAB over shared memory.

A custom interactive MATLAB graphical user interface (GUI), shown in Fig. 3, is written to serve as the single user interface for our framework. The MATLAB GUI interfaces with TI mmWave Studio [28] to control the hardware setup and initializes the UDP interface, bypassing the need for user setup outside our GUI. The radar continuously captures and streams data into MATLAB using the fully-integrated implementation. While MATLAB does not offer the computational speed necessary for real-time system implementation, it is capable of completing the data capture, signal processing, deep learning, visualization, and signal output at around 250 Hz, from our experimentation. In the early prototyping phase, we consider this throughput sufficient for investigating the performance of the super-resolution tracking algorithms and a simple musical interface.

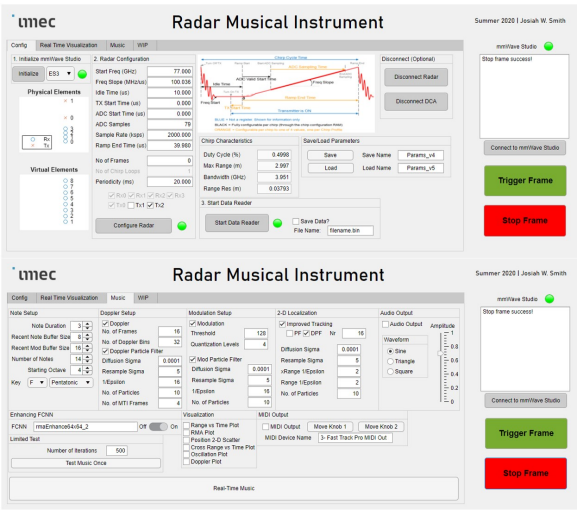

Fig. 3. Interactive MATLAB GUI: device setup and music generation pages.

Using our proposed methods, the software extracts high-resolution spatiotemporal features of the user's hand and maps them to corresponding output using either a built-in audio output tool or the included MIDI output. The custom MATLAB GUI provides an accessible option for investigating and demonstrating our methods as well as an open-source platform to stimulate further collaborative investigation by the multimedia and radar communities.

As previously mentioned, the primary mechanism to control the output of the proposed musical interface is the range (z-position) of the user's hand. Using the built-in audio output tool and the MIDI output, the range of the user's hand controls the note selection directly. Unlike the Theremin, which allows for continuous note selection, our interface quantizes the user input into predefined subregions corresponding to notes defined by the user. The subregions and allowed notes can be programmed by the user in the interactive MATLAB GUI. To play the desired note, the user must move their hand vertically to the position corresponding to that note. Similarly, the secondary parameters, cross-range oscillation, and velocity can be adjusted by the user by oscillating their hand back-and-forth in the y-direction or moving to the next note with a high or low velocity. The built-in audio output tool employs the cross-range oscillation to control a vibrato effect (low-frequency modulation of the audio signal). Thus, using this tool, the user can select the desired note by varying the range and perform vibrato at a desired rate by oscillating their hand at the same rate. Alternatively, the MIDI output tool provides the cross-range oscillation and velocity as MIDI parameters to be specified by the user in a virtual instrument environment connected to the MIDI output of our musical interface. Hence, our proposed algorithms are implemented to operate similarly to a MIDI keyboard with the hand range controlling the note selection and cross-range oscillation and velocity acting as MIDI parameters for the user to assign.

C. Simple Feature Extraction and Tracking Algorithm Signal Processing Chain

The signal processing chain for the simple feature extraction and tracking method is shown in Fig. 4. The beat signal is loaded into MATLAB where the preprocessing discussed in the previous section is performed (RMA and peak finding) and the user inputs (2-D location and velocity) are converted into audio or MIDI output by extracting the spatiotemporal features using (9) and (11). In this article, the location and velocity of the user's hand are used for musical gestural interface; however, our novel algorithms can easily be applied to many different HCI applications and even for 3-D localization, provided a sufficient 2-D array. The reconstructed RMA image and raw feature extracted by the classical techniques can be utilized by the particle filter algorithm and super-resolution FCNN to improve the tracking performance.

D. Super-Resolution Framework - Training FCNN and Implementing Particle Filter Algorithm

To implement our super-resolution feature extraction and tracking framework, the super-resolution FCNN must be first

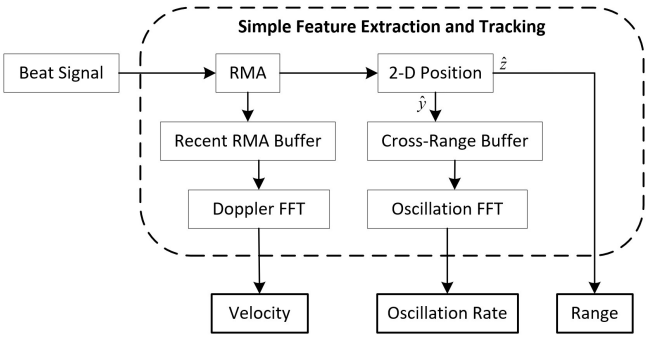

Fig. 4. Simple tracking signal processing chain. After RMA is performed on beat signal, features are extracted directly from raw RMA image.

trained. The enhancement FCNN is trained using both real data from a human hand and simulated data corrupted by additive real radar noise. The FCNN is trained using 65536 simulated and 23040 real human hand RMA images as the input and output images with $\sigma_y = \sigma_z = 1$ mm resulting in cross-range and range resolutions of 1.18 mm. Each simulated sample is generated at a random location in the ROI $y \in [-0.1, 0.1]$, $z \in [0.1, 0.5]$. The synthetic data cover the entire ROI allowing the network to generalize well to location while learning the non-idealities of the imaging scheme. 512 samples of a real hand are collected at each of the 45 locations throughout the ROI as shown in Fig. 5. For both the synthetic samples and real human samples, corresponding ground truth images are generated using (14) and used as training labels. Thus, the training set is comprised of features consisting of real and simulated data and labels consisting of the ideal expected response at each known location.

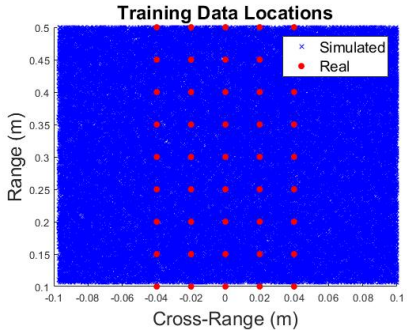

Fig. 5. Locations of the training data used to train the enhancement FCNN. Real data (red) are collected by keeping the hand static at known locations. Simulated data (blue) are generated by choosing locations randomly from the continuous ROI.

The architecture of the proposed enhancement FCNN is shown in Fig. 6. The network consists of four convolution layers of decreasing kernel size each followed by a nonlinear Rectified Linear Unit (ReLU) layer. Each convolutional layer is zero-padded such that the output is identical in size to the input. Training the network for 100 epochs takes 5 hours on a machine with a single NVIDIA GTX1080TI graphics card. Other network architectures and training durations are investigated, but this combination yields high performance while offering real-time efficiency.

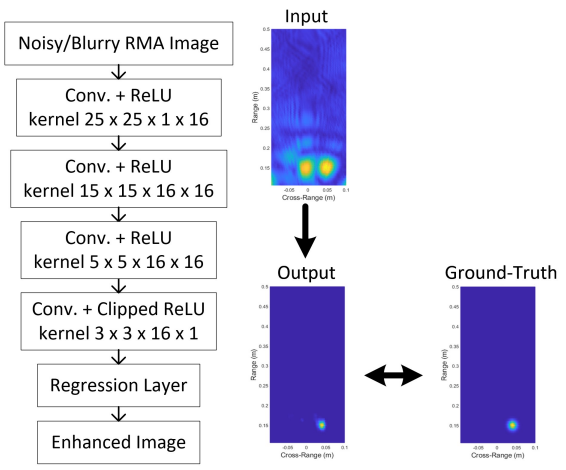

Fig. 6. Architecture of the enhancement FCNN. The selected kernel and layer sizes are capable of adequately learning the non-ideal shape of the distorted RMA image while maintaining high computational efficiency for real-time implementation.

Once the super-resolution FCNN has been trained by the proposed technique, our novel tracking algorithm can be implemented using the particle filter discussed previously. The Doppler-corroborated particle filter is employed to track the position of the hand in the y-z plane and two additional particle filters are used to track the Doppler velocity and cross-range oscillation. The entire signal processing chain for the enhanced feature extraction and tracking method is shown in Fig. 7. The spatiotemporal features are outputted from the algorithm and can be used for many tracking applications. Additionally, if the 2-D location of the hand is desired over the range and cross-range oscillation rate, the algorithm can be easily adapted to output the desired spatial features.

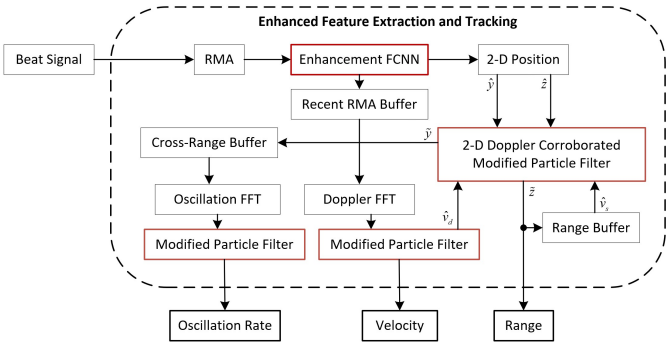

Fig. 7. Enhanced tracking signal processing chain. Key elements to the enhanced methods are highlighted in red.

V. Spatiotemporal Feature Extraction and Tracking Results

In this section, we overview the results of our novel tracking and feature enhancement algorithms beginning with the simple, classical techniques and comparing against our proposed methods. Our enhanced tracking regime demonstrates considerable performance improvement compared with the traditional methods and allows for robust super-resolution tracking on a small radar platform unattainable by existing methods.

A. Ground Truth - Ideal Motion Profile

To verify the feature estimation techniques, a virtual prototyping approach is adopted. A point target is simulated in motion with y-z location and velocity shown in Fig. 8 using (1). This ideal motion profile is employed to compare the tracking performance of our proposed methods to the traditional techniques. Real noise collected from the radar with an empty scene is added to each synthetic beat signal as

$$\tilde{s}(y_T, y_R, k) = \frac{p}{R_T R_R} e^{jk(R_T+R_R)} + \alpha\tilde{\omega}(y_T, y_R, k), \quad (15)$$

where $\tilde{\omega}$ is a complex-valued noise sample corrupting the amplitude and phase of the ideal simulated beat signal and α controls the SNR.

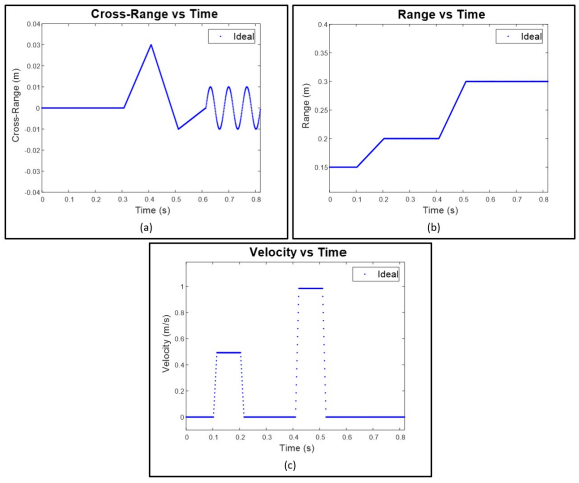

Fig. 8. Ideal motion profile of the target in the (a) cross-range and (b) range directions as well as the (c) range velocity profile against time.

The motion profile shown in Fig. 8 shows the ideal range (z), cross-range (y), and velocity (v) of the target. The motion profile includes independent and joint movement in the range and cross-range domains in addition to sinusoidal cross-range oscillation. For our simulations, 4096 time samples are generated using $p \in [0.5, 1]$ to simulate the variance in the hand's empirical radar cross-section (RCS) as observed from prior hand data and $\alpha \in [1, 3]$ to vary the SNR among samples. Values for p and α are selected randomly within the specified intervals for each time sample and provide a level of stochastic realism to the simulated data.

B. Classical Spatiotemporal Imaging Results

First, the simple tracking methods discussed in Section III-A are implemented to provide baseline performance metrics. The signal processing chain shown in Fig. 4 is performed, extracting the spatiotemporal features. At each iteration, the features are extracted directly from the raw RMA images and are therefore prone to erratic behavior.

Fig. 9 shows the features estimated from the data generated by (15) using the simple methods. The real radar noise and

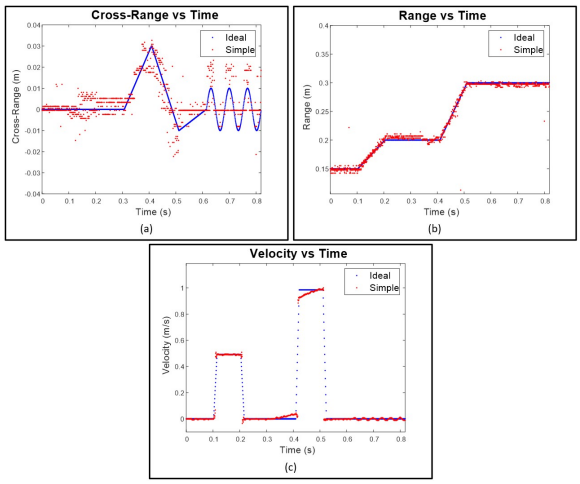

Fig. 9. Motion profile using simple features extraction techniques on each frame for every time step (red) compared with the ideal motion and velocity profiles (blue). The (a) cross-range and (b) range are measured directly from the peak of the RMA image of each frame and the (c) velocity is measured using the Doppler FFT of the raw RMA images using (10) and (11).

varying reflectivity result in outliers and errors in the estimated location and velocity of the target, particularly in the cross-range domain. Without more robust feature extraction and tracking techniques, the performance leaves much to be desired. In the following sections, the performance of the simple tracking methods is quantitatively compared to the enhanced tracking methods and design considerations are discussed.

C. FCNN-Based Super-Resolution Tracking Results

Assuming the motion profile in Fig. 8, our proposed particle filter algorithm is employed in an attempt to more robustly track the 2-D position and Doppler velocity of the target across time, improving the user's control over the interface significantly[1].

First, the particle filter algorithm (PF) without Doppler corroboration is implemented using the data in Fig. 9 as elements of the noisy measurement vector $\boldsymbol{r}$. The PF reduces the effect of the noise on the position estimation and improves the spatiotemporal tracking performance as shown in Fig. 10. The cross-range position tracking is most improved compared to the traditional methods. Next, the Doppler-corroborated particle filter (DPF) is applied to the same set of data further improving the estimation of the range. The outliers in Fig. 10b are mitigated by the DPF in Fig. 10d because the outlying samples result in a sample velocity $\hat{v}_s$ contradicted by the Doppler velocity $\hat{v}_d$ and are weighted as unimportant in the resampling process. The DPF algorithm improves the user experience of our interface by providing a robust, consistent tracking algorithm to smoothly estimate the 2-D position and spatiotemporal signatures of the user's hand. However, the PF and DPF can be further improved by implementing the proposed enhancement FCNN.

After the super-resolution FCNN is trained using the technique discussed in Section IV-D, a validation dataset of identical size to the training set is collected. Fig. 11 shows images enhanced by the enhancement FCNN demonstrating the robustness of the network. Figs. 11a and 11b show simulated point targets enhanced by the FCNN resulting in localization super-resolution. Fig. 11c is an RMA image reconstructed

[1]Supplemental material for the reader can be downloaded at http://ieeexplore.org/

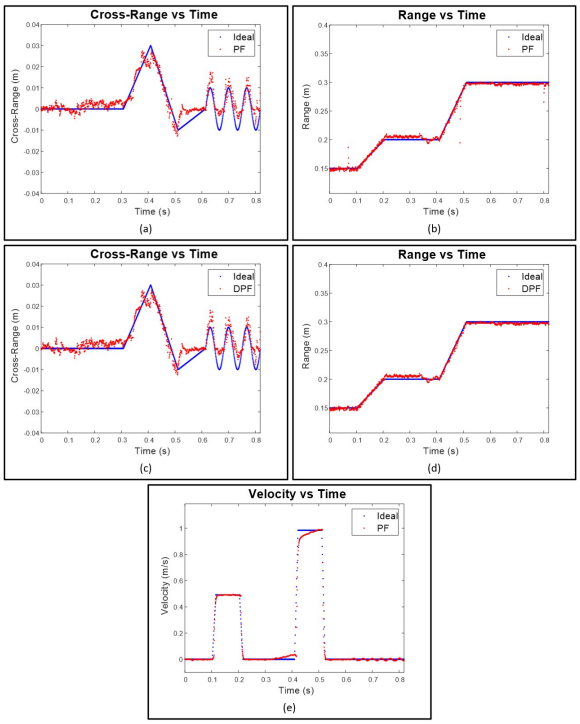

Fig. 10. The particle filter (PF) and Doppler-corroborated particle filter (DPF) algorithms employed for robust spatiotemporal tracking of the simulated gestures through time: improved tracking of the (a) cross-range and (b) range versus time using the PF, (c) cross-range and (d) range versus time using the DPF with $N_z = 16$, and (e) Doppler velocity versus time using a PF approach.

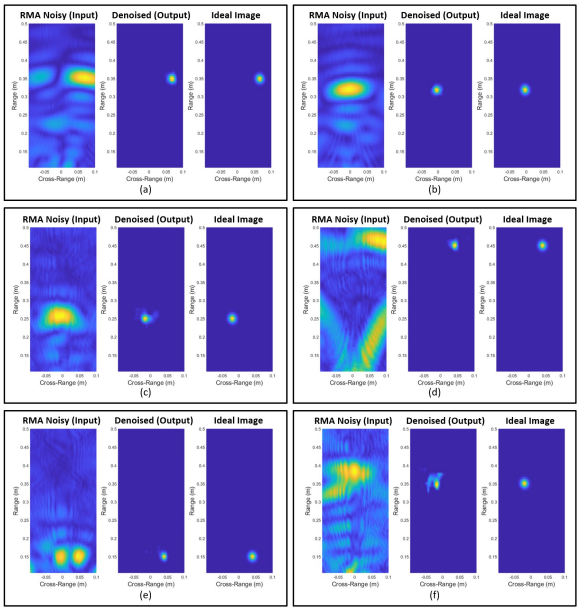

Fig. 11. Enhancement FCNN applied to simulated (a,b) and real hand (c-f) RMA images for image enhancement and improved localization.

from a real hand capture close to the middle of the cross-range domain. The 2-D position of the hand is accurately located compared with the ideal image. Similarly, Figs. 11d-11f demonstrate the network's ability to enhance images degraded by small hand RCS in comparison to noise, ghosting due to non-ideal beam patterns, ambient and device noise, and other non-idealities. The proposed enhancement FCNN simultaneously enables localization super-resolution and overcomes device and environment issues. Hence, the features extracted from the enhanced images are much improved compared to the raw RMA images before the FCNN and result in superior tracking performance.

TABLE I
SIMPLE VS ENHANCED LOCALIZATION RMSE

	y (m)	z (m)
Simple	0.0154	0.023
Enhanced	0.0085	0.0083

To quantitatively compare the localization improvement of the enhancement FCNN compared to the simple method, the RMSE in the range and cross-range position are computed on the validation dataset using the two techniques and shown in Table I. The enhancement FCNN improves both the resolution of the RMA images and the localization accuracy for both simulated and real data.

Applying the FCNN and DPF (FCNN-DPF) to the raw data following the ideal motion profile in Fig, 8, yields further tracking improvement over the DPF alone. Fig. 12 demonstrates the tracking performance of the FCNN-DPF on the same data as the previous tracking examples. Applying the FCNN-DPF, the range and cross-range tracking of the target is nearly identical to the ideal motion profile and an improvement in the velocity estimation. Using the identical sporadic data resulting in the poorly estimated cross-range positions in Fig. 9a, the FCNN-DPF yields an estimation nearly identical to the ideal motion profile. Similarly, the cross-range estimates in Fig. 10a and Fig. 10c are outperformed by the FCNN-DPF in Fig. 12a. Compared to the classical techniques and PF/DPF alone, the localization performance of the FCNN-DPF is considerably superior.

Further, the FCNN improves the Doppler estimation robustness. As shown in Fig. 13, the Doppler spectrum SNR is improved when the Doppler processing is performed on the enhanced RMA images as compared to Doppler processing on the raw RMA images. Hence, the enhancement network improves the reliability of the Doppler velocity estimation aiding spatiotemporal tracking.

VI. DISCUSSION AND FUTURE WORK

To quantitatively compare the tracking performance of the various proposed methods, 4096 unique motion profiles are generated and corresponding tracking RMSE is computed for the cross-range, range, and velocity. Displayed in Table II, the RMSE for the cross-range (y), range (z), and velocity (v) improve with the novel algorithms proposed in this paper.

As expected, the baseline simple method yields the greatest error for all three features. Comparing PF and DPF, the cross-

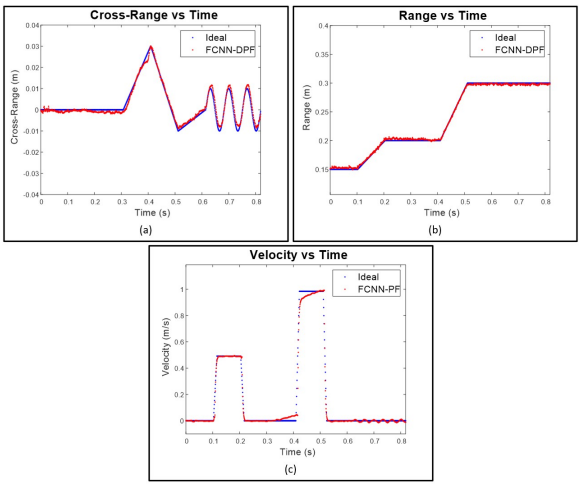

Fig. 12. The FCNN enhanced Doppler-corroborated modified particle filter algorithm.

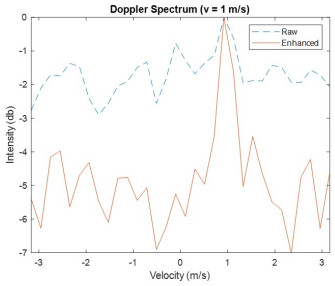

Fig. 13. Comparison of the Doppler velocity spectrum when the Doppler FFT and video pulse integration steps are performed on the raw RMA images compared to the enhanced RMA images. The simulated data contains 128 frames and uses $\alpha = 3$ for every capture to simulate a low SNR scenario.

range and velocity RMSE are identical between the two techniques but the range RMSE is improved due to the dynamic weighting technique. The FCNN alone outperforms the simple method but can be improved by including the PF and DPF after image enhancement. Finally, the FCNN-PF and FCNN-DPF yield identical results for the cross-range and velocity RMSE, as expected, but significant improvement can be noted in the range error. The results in Table II demonstrate the considerably superior tracking performance of the enhanced tracking methods, namely the FCNN-DPF, compared with the simple tracking method. The performance gain realized by implementing the super-resolution FCNN demonstrates the ability of the network to learn the system noise and ambiguities during the training phase using both real and synthetic data.

The average latency of each method, $\bar{\tau}$, is measured as the time duration between the new sample being captured and the estimation process being completed on that sample. The resulting estimates are streamed across the MIDI port or sent to the built-in audio signal generation tool. Additional latency contributed by the subsequent synthesis engine is highly dependent on the software used and device under test; thus, it is not considered as part of the latency due to our methods.

The enhanced tracking methods outperform the exist-

TABLE II
AVERAGE RMSE FOR TRACKING METHODS

	y (mm)	z (mm)	v (mm/s)	$\bar{\tau}$ (ms)
Simple	7.86	22.0	72.4	2.29
PF	5.27	13.6	52.9	2.36
DPF	5.27	6.85	52.9	2.41
FCNN	7.74	12.3	58.4	2.67
FCNN-PF	3.70	7.44	44.5	3.92
FCNN-DPF	3.70	3.07	44.5	3.96

ing techniques in localization resolution, Doppler spectrum SNR, and tracking accuracy; however, there are some necessary trade-offs for this performance gain. The novel super-resolution FCNN yields noteworthy resolution improvement over the theoretical bounds. In ideal conditions, the cross-range and range resolutions of our system are bounded by $\delta_y = 7.5$ cm and $\delta_z = 3.75$ cm, respectively. Using the combination of real hand data and synthetic data used in Table I, the spatial resolution in each direction is computed empirically as $\delta_y = 2.3$ mm and $\delta_z = 1.96$ mm. On the other hand, the effectiveness of the enhancement FCNN is limited by the training set. Since the FCNN is only trained on images within the expected region, extending the ROI outside of the trained region results in performance degradation. If the ROI is changed, the FCNN should be retrained accordingly. In contrast, the simple methods are highly flexible but cannot compete with the performance of the enhancement techniques. However, we have studied the limitations of the particular TI mmWave radar device and found that if the hand is placed outside the ROI defined in the previous section, it will not be detected. Due to device SNR and beamwidth, for most hand sizes, the reflections back to the radar will not be strong enough for detection. Additionally, we have tested the proposed FCNN in smaller ROIs and found similar results without retraining. For other array topologies, the proposed methods can be easily applied, although the FCNN will need to be trained accordingly.

While the Doppler-corroborated particle filter improves the tracking robustness, it requires a high throughput framework to function properly. Since the DPF relies on accurate Doppler velocity estimation, the pulse repetition interval PRI, T_{PRI}, must be sufficiently small such that high hand velocities are within the resolvable range. The PRI is impacted most significantly by the time per iteration of the signal processing chain. Given our framework is currently released in the prototyping stage as a MATLAB program, the latency performance does not match that of a real-time implementation on a more efficient embedded device. Hence, typical throughput times limit the PRI to around 4 ms. At $T_{PRI} = 4$ ms using a 77 GHz device, the maximum resolvable velocity is 0.24 m/s. With this limitation, some rapid movements at high velocities may result in Doppler spectrum aliasing.

While the software package presented in this article serves as a framework for demonstrating and prototyping the proposed tracking and super-resolution algorithms, the inherent latency of the signal processing steps is a key issue in HCI and must be addressed. In our research, the largest contributor of latency in our proposed system is the hand-off between the radar device and MATLAB over UDP and shared memory, at an average of 1.93 ms. Rather than streaming to data MATLAB, a real-time solution can be implemented on the TI radar device's built-in DSP, thus providing a more efficient throughput as the DSP has direct access to the samples as they are taken. Additionally, several steps in the signal processing chain will increase in efficiency with an embedded solution. Employing small window sizes, $N_z = 16$ and the number of FFT spatial points is 64, the DPF and FFT computation times can be further reduced compared to the relatively inefficient MATLAB implementation. We would also like to note that a significant decrease in latency was achieved by optimizing the implementation using GPU accelerated coding. A similar approach could be taken on an embedded solution leveraging the highly parallelizable nature of many of the steps in the signal processing chain (FFT, CNN, Gaussian distribution computation). Comparing the computational efficiency among the algorithms, the latency cost for the more robust algorithms is insignificant in proportion to the performance gain, even in the MATLAB implementation. In latency tests, the average response time using the FCNN-DPF was 3.96 ms from user input to MIDI signaling. While most MIDI interfaces outperform this metric, we believe our framework demonstrates a competitive throughput cycle time compared to existing technology and can be further improved by a more efficient implementation.

Hand-tracking using a mmWave radar has both advantages and drawbacks compared with other sensing regimes. In this article, we employ a single radar to develop and demonstrate robust tracking algorithms for mmWave devices. While the best performance is likely achieved through a sensor fusion technique, a radar-based implementation may be optimal if privacy is a concern using optical cameras or issues such as occlusion and lighting conditions must be taken into account. Compared to optical and RGB+D solutions, mmWave is more versatile and reliable, operating well under occlusion, in any temperature or lighting environment, and offers precise depth information of the entire scene. For a musical interface, these advantages may not be often fully realized; however, the novel tracking methods proposed in this article are applicable for many HCI applications. On the other hand, mmWave sensors cannot meet the performance of optical solutions when it comes to cross-range resolution due to the limited aperture size, making multi-object and finger tracking much more challenging. As such, many applications in HCI, computer vision, automated driving, etc. employ radar (and lidar) and optical imaging devices with sensor fusion algorithms to achieve further improved performance at an increased cost. For these applications, our proposed algorithms can aid in sensor fusion by significantly increasing the performance contribution from the radar sensors.

Several alternatives exist to mmWave radar sensing, namely wearable, handheld, and optical devices. Wearable and handheld sensing solutions offer highly precise spatiotemporal features but are often not preferable compared to contactless sensors [29], [30]. In terms of cost, mmWave radar devices are in the same price bracket as the popular Kinect and Leap Motion optical sensors on the order of $\$100 - \$200$. Attempts

using multiple RGB cameras [31], [32] show promising results; however, a single device is much preferred due to the cumbersome nature of multi-camera systems. Single RGB+D solutions have been proposed using generative pose tracking [33], [34] and learning-based generative pose tracking [35], [36]. However, all of these methods suffer tremendously under occlusion or scene clutter, both of which can be overcome using mmWave radar. Some deep learning-oriented solutions have shown quite promising results [37], [38], but constructing a sufficient dataset for meaningful supervised training remains a challenge.

Our proposed interface tracks the 2-D position and velocity of the user's hand to control note selection and two user-selected parameters, a marked improvement over the prior work on mmWave radar using the Google Soli tracking only 1-D range for parameter control [16]. However, optical solutions enable tracking of both hands [3], [7], [9], [33]–[38] or hand and finger position [8], [12] for even finer musical control, with some scenario-specific drawbacks. As radar technology improves and larger apertures become widely available, tracking individual fingers will become increasingly plausible and could yield comparable or superior results to optical solutions due to superior depth resolution.

Compared to prior work on hand-tracking with mmWave devices, our proposed methods yield competitive results. Past work using radar devices achieves, at best, an average range tracking error of 2 cm on human hand localization [17]. Our enhanced tracking technique yields a mean range tracking error of 1.89 mm, improving tracking by more than a factor of ten. In [39], a 4 GHz bandwidth mmWave sensor achieves a 2-D position RMSE of 1.16 mm tracking a thumb, at distances closer than 10 cm. Comparatively, our enhanced tracking technique tracks a human hand across much larger distances and still achieves a competitive 2-D position RMSE of 3.4 mm. At the time of this paper, we are not aware of any other prior work on hand-tracking using mmWave devices. To our knowledge, the system proposed in this paper offers unprecedented hand gesture tracking performance using a single mmWave sensor.

The most direct musical interface comparison to our framework, is the Theremin, as both are controlled by the hand's proximity to the sensor. The pitch of the Theremin is controlled continuously by the hand's vertical location, whereas our interface tracks the range of the hand digitally and selects a note from the user-defined scale. While the Theremin uses two antennas, one for volume control and the other for pitch control, a total of two degrees of freedom, our framework offers three degrees of freedom (range, cross-range, and velocity), thus providing three controllable parameters. As previously mentioned, the musical interface promoted in this article supports Theremin-like gestures for note selection and parameter control. However, high-velocity percussive gestures could be implemented using our high-fidelity tracking algorithms, with some limitations. Small values of the weighting factor, $\boldsymbol{A}$, in the particle filter algorithm can result an excessively smoothed and overly damped system limiting the ability of the system to track sudden movements. Depending on the desired application, finely tuning this parameter is essential for enabling proper gestural control. Our proposed interface is an evolved Theremin, utilizing a modern mmWave sensor for precise tracking in 2-D space (expansion to 3-D can be easily implemented with the proper hardware). In contrast to a Theremin, our musical interface is significantly less effortful in note selection, allowing simple and intuitive inclusion of the additional parameter controls and increasing accessibility to the user-base. One of the authors is a skilled guitar and violin instrumentalist with a background in electronic music production. From the perspective of an experienced musician, the proposed methods offer an elegant new musical interface capable of generating unique phrases previously only possible via manual transcription and provides the musician a sufficient and consistent level of control.

For future work, several promising routes are left to be explored. First, further development can be explored by implementing our proposed methods onto a real-time embedded platform. Additionally, using multiple MIMO radars or a larger MIMO array, a multiple-hand and individual finger tracking interface can be investigated, thus further extending the application space of our robust tracking methods. Finally, our novel super-resolution tracking algorithms can easily be adapted to offer an elegant, efficient solution to a host of acute hand-tracking problems in the HCI domain and even employed in sensor-fusion systems.

VII. Conclusion

Our FCNN-based super-resolution framework successfully demonstrates the viability of acute human hand-tracking for HCI using mmWave sensors. We validated and implemented our spatiotemporal signal processing algorithms and robust tracking algorithms in the form of a contactless musical interface; however, this article also serves to demonstrate the broad effectiveness of mmWave technology for a multitude of near-field acute hand-tracking applications. First, simple feature extraction and tracking methods were introduced, followed by an enhanced approach leveraging the Doppler-corroborated particle filter algorithm and enhancement FCNN to achieve robust tracking and super-resolution in a non-ideal imaging scenario. The methods are compared demonstrating noticeable improvement using the FCNN-DPF over the classical techniques. Additionally, our work offers competitive tracking estimation and localization performance compared to prior methods in the literature for both mmWave and optical implementations. Our entire software implementation and real-time radar interface platform are freely available at request. The novel FCNN-based super-resolution and tracking algorithms presented in this article offer an elegant solution to many contactless HCI problems.

Acknowledgment

The first author's work was supported by the imec USA summer internship program. We would like to extend thanks to Dr. Gonzalo Vaca Castano for his insights in developing the particle filter algorithm and computer vision approach.

REFERENCES

[1] T. Winkler, "Making motion musical: Gesture mapping strategies for interactive computer music," in *Proc. Int. Computer Music Conf.*, Banff, Canada, 1995, pp. 261–264.

[2] K. D. Skeldon, L. M. Reid, V. McInally, B. Dougan, and C. Fulton, "Physics of the theremin," *American Journal of Physics*, vol. 66, no. 11, pp. 945–955, 1998.

[3] R. Polfreman, "Multi-modal instrument: towards a platform for comparative controller evaluation," in *Proc. Int. Computer Music Conf.* Proc. Int. Computer Music Conf., July 2011, pp. 147–150. [Online]. Available: https://eprints.soton.ac.uk/353226/

[4] S. Trail, M. Dean, G. Odowichuk, T. F. Tavares, P. F. Driessen, W. A. Schloss, and G. Tzanetakis, "Non-invasive sensing and gesture control for pitched percussion hyper-instruments using the kinect." in *Proc. Int. Conf. New Interfaces for Musical Expression*, 2012.

[5] S. Sentürk, S. W. Lee, A. Sastry, A. Daruwalla, and G. Weinberg, "Crossole: A gestural interface for composition, improvisation and performance using kinect." in *Proc. Int. Conf. New Interfaces for Musical Expression*, 2012.

[6] R. Schramm, C. R. Jung, and E. R. Miranda, "Dynamic time warping for music conducting gestures evaluation," *IEEE Trans. on Multimedia*, vol. 17, no. 2, pp. 243–255, 2015.

[7] A. R. Jensenius, "Kinectofon: Performing with shapes in planes," in *Proc. Int. Conf. on New Interfaces for Musical Expression*, Daejeon, Korea, 2013, pp. 196–197.

[8] J. Han and N. Gold, "Lessons learned in exploring the leap motion™ sensor for gesture-based instrument design," in *Proc. Int. Conf. on New Interfaces for Musical Expression*. London, United Kingdom: Goldsmiths University of London, 2014, pp. 371–374.

[9] L. Hantrakul and K. Kaczmarek, "Implementations of the leap motion in sound synthesis, effects modulation and assistive performance tools." in *Proc. Int. Conf. on New Interfaces for Musical Expression*, London, United Kingdom, 2014.

[10] D. Brown, N. Renney, A. Stark, C. Nash, and T. Mitchell, "Leimu: Gloveless music interaction using a wrist mounted leap motion," in *Proc. Int. Conf. on New Interfaces for Musical Expression*, Brisbane, Austrailia, 2016, pp. 300–304.

[11] A. Tindale, A. Kapur, and G. Tzanetakis, "Training surrogate sensors in musical gesture acquisition systems," *IEEE Trans. on Multimedia*, vol. 13, no. 1, pp. 50–59, 2011.

[12] O. Nieto and D. Shasha, "Hand gesture recognition in mobile devices: Enhancing the musical experience," *Proc. Computer Music Multidisciplinary Research*, vol. 13, 2013.

[13] M. Akbari and H. Cheng, "Real-time piano music transcription based on computer vision," *IEEE Trans. on Multimedia*, vol. 17, no. 12, pp. 2113–2121, 2015.

[14] J. W. Smith, S. Thiagarajan, R. Willis, Y. Makris, and M. Torlak, "Improved static hand gesture classification on deep convolutional neural networks using novel sterile training technique," *IEEE Access*, vol. 9, pp. 10 893–10 902, 2021.

[15] S. Skaria, A. Al-Hourani, M. Lech, and R. J. Evans, "Hand-gesture recognition using two-antenna doppler radar with deep convolutional neural networks," *IEEE Sensors Journal*, vol. 19, no. 8, pp. 3041–3048, 2019.

[16] F. Bernardo, N. Arner, and P. Batchelor, "O soli mio: exploring millimeter wave radar for musical interaction." in *Proc. Int. Conf. on New Interfaces for Musical Expression*, vol. 17, 2017, pp. 283–286.

[17] K. Joshi, D. Bharadia, M. Kotaru, and S. Katti, "Wideo: Fine-grained device-free motion tracing using rf backscatter," in *Proc. USENIX Symposium on Networked Systems Design and Implementation*, 2015, pp. 189–204.

[18] Y. Dai, T. Jin, Y. Song, H. Du, and D. Zhao, "Cnn-based multiple-input multiple-output radar image enhancement method," *The Journal of Engineering*, vol. 2019, no. 20, pp. 6840–6844, 2019.

[19] Y. Sun, X. Liang, H. Fan, M. Imran, and H. Heidari, "Visual hand tracking on depth image using 2-d matched filter," in *Proc. UK/China Emerging Technologies*, Aug. 21-22, 2019, Glasgow, United Kingdom, pp. 1–4.

[20] S. Rao, "Intro to mmwave sensing : Fmcw radars," Jul 2020. [Online]. Available: https://training.ti.com/node/1139153

[21] J. W. Smith, M. E. Yanik, and M. Torlak, "Near-field mimo-isar millimeter-wave imaging," in *Proc. IEEE Radar Conf.*, 2020, pp. 1–6.

[22] M. E. Yanik and M. Torlak, "Near-field mimo-sar millimeter-wave imaging with sparsely sampled aperture data," *IEEE Access*, vol. 7, pp. 31 801–31 819, 2019.

[23] M. E. Yanik, D. Wang, and M. Torlak, "Development and demonstration of mimo-sar mmwave imaging testbeds," *IEEE Access*, vol. 8, pp. 126 019–126 038, 2020.

[24] V. Winkler, "Range doppler detection for automotive fmcw radars," in *Proc. European Radar Conf.*, Oct. 10-12, 2007, Munich, Germany, pp. 166–169.

[25] J. Kim, J. Chun, and S. Song, "Joint range and angle estimation for fmcw mimo radar and its application," *arXiv:1811.06715*, 2018.

[26] J. García, A. Gardel, I. Bravo, J. L. Lázaro, and M. Martínez, "Tracking people motion based on extended condensation algorithm," *IEEE Trans. on Systems, Man, and Cybernetics: Systems*, vol. 43, no. 3, pp. 606–618, 2013.

[27] J. Gao, B. Deng, Y. Qin, H. Wang, and X. Li, "Enhanced radar imaging using a complex-valued convolutional neural network," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 1, pp. 35–39, 2019.

[28] "Texas instruments mmwave studio." [Online]. Available: https://www.ti.com/tool/MMWAVE-STUDIO

[29] L. Pardue and W. Sebastian, "Hand-controller for combined tactile control and motion tracking." in *Proc. Int. Conf. on New Interfaces for Musical Expression*, 2013, pp. 90–93.

[30] P. Neto, J. N. Pires, and A. P. Moreira, "High-level programming and control for industrial robotics: using a hand-held accelerometer-based input device for gesture and posture recognition," *Industrial Robot*, vol. 37, no. 2, pp. 137–147, 2010.

[31] L. Ballan, A. Taneja, J. Gall, L. Van Gool, and M. Pollefeys, "Motion capture of hands in action using discriminative salient points," in *Proc. European Conf. on Computer Vision*. Springer, 2012, pp. 640–653.

[32] S. Sridhar, A. Oulasvirta, and C. Theobalt, "Interactive markerless articulated hand motion tracking using rgb and depth data," in *Proc. IEEE Int. Conf. on Computer Vision*, 2013, pp. 2456–2463.

[33] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Efficient model-based 3d tracking of hand articulations using kinect." in *Proc. Brit. Mach. Vision Conf*, vol. 1, no. 2, 2011, pp. 101.1–101.11.

[34] D. Tang, J. Taylor, P. Kohli, C. Keskin, T.-K. Kim, and J. Shotton, "Opening the black box: Hierarchical sampling optimization for estimating human hand pose," in *Proc. IEEE Int. Conf. on Computer Vision*, 2015, pp. 3325–3333.

[35] S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt, "Fast and robust hand tracking using detection-guided optimization," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2015, pp. 3213–3221.

[36] J. Taylor, L. Bordeaux, T. Cashman, B. Corish, C. Keskin, T. Sharp, E. Soto, D. Sweeney, J. Valentin, B. Luff *et al.*, "Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences," *ACM Trans. on Graphics*, vol. 35, no. 4, pp. 1–12, 2016.

[37] J. Tompson, M. Stein, Y. Lecun, and K. Perlin, "Real-time continuous pose recovery of human hands using convolutional networks," *ACM Trans. on Graphics*, vol. 33, no. 5, pp. 1–10, 2014.

[38] Q. Ye, S. Yuan, and T.-K. Kim, "Spatial attention deep net with partial pso for hierarchical hybrid hand pose estimation," in *Proc. European Conf. on Computer Vision*. Springer, 2016, pp. 346–361.

[39] Z. Li, Z. Lei, A. Yan, E. Solovey, and K. Pahlavan, "Thumouse: A micro-gesture cursor input through mmwave radar-based interaction," in *Proc. IEEE Int. Conf. on Consumer Electronics*, Jan. 4-6, 2020, Las Vegas, NV, USA, pp. 1–9.