Radar Musical Instrument - A Spatiotemporal Real-Time mmWave Sensor for Contactless Human-Computer Interaction

Josiah 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]Computational Imaging, imec USA, FL, United States

***Abstract*—Millimeter-wave (mmWave) radar sensing is transforming many applications that have traditionally required different modes of sensing, as exemplified by self-driving cars, vital signs monitoring, fall detection, occupancy detection, and many more. Human-computer interaction (HCI) can benefit from the use of mmWave radars because of the fine depth and cross-range resolution of such devices, enabling accurate tracking of user-performed actions in space. Additionally, signatures embedded in the return signal from frequency-modulated continuous-wave (FMCW) radars contain spatiotemporal features that can be extracted in real time to perform accurate position and pose tracking for HCI. In this paper, we propose and demonstrate a novel real-time mmWave interface that leverages spatiotemporal information from a multiple-input-multiple-output (MIMO)-FMCW radar to create a new musical interface (NMI) controllable by specific hand positions and motions. After constructing the necessary real-time framework, a simple signal processing chain and feature extraction method is presented and subsequently extended to an enhanced tracking technique employing novel localization algorithms and deep-learning-based spatiotemporal enhancement. The novel system we propose in this paper allows for real-time human-computer interaction to create a new musical interface controlled solely by the precise tracking of the musician's hand.**

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

I. Introduction

Radar perception for human-computer interaction on multiple-input-multiple-output (MIMO) frequency modulated continuous wave (FMCW) 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, ultra-wideband FMCW devices enable depth resolution on the order of centimeters and compact MIMO radars allow for task-enabling cross-range resolutions on a small scale. As a result, precise spatial information of a target scene can be easily acquired from such imaging devices at a low cost.

mmWave sensing solutions are relatively modern technology, but some of the earliest electronic interfaces, for any application, 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. Polfreman uses the Kinect to track the 3-D position of both hands of a standing performer to construct a multi-modal instrument [3]. Trail *et al.* present a pitched percussion hyperinstrument to track the tips of two mallets simultaneously with the Kinect [4]. Crossole, designed by Senturk *et al.*, is a Kinect-based metainstrument visualizing 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 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 in the hand 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 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" by learning and monitoring for gestures performed by the musician [11]. In [12], a musical interface is constructed using exclusively a portable RGB camera to recognize hand gestures. 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 and demonstrate the broad application space of mmWave imagers and advanced spatiotemporal algorithms. However, little work has been done towards gestural musical interfaces on mmWave radar sensors. Even though extensive research exists on static and dynamic gesture recognition using 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 Radar Musical Instrument presented in this paper 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 image enhancement technique. Additionally, a spatiotemporal tracking algorithm is presented with a novel Doppler-corroboration extension further improving tracking robustness. Compared to prior work on gesture tracking using optical solutions [3], [7]–[9], [12], [18] or optical-mmWave [19] sensor fusion, our approach addresses the issue of hand gesture tracking using a singular mmWave sensor. The methods presented in this work offer a novel approach to gesture tracking 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 being made available by request. To our knowledge, the Radar Musical Instrument 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. In section II, a brief overview of relevant music theory is provided for a requisite understanding of the applicable musical topics discussed throughout the paper. Section III provides an overview of the MIMO-FMCW radar signal model and feature extraction methods. In section IV, the novel Radar Musical Instrument is introduced, presenting two robust tracking algorithms and estimation techniques, the results of which are shown in section V. Section VI contains a discussion of the performance, design constraints, and distinct advantages of the two tracking methods in sections IV-B and IV-C, followed finally by conclusions.

II. Music Theory for Engineers

A. Notes, Pitch, Frets, and Vibrato

The fundamental unit of music is the note and every instrument and interface requires the musician to input or select the note differently. Notes are primarily distinguished by the human brain according to their pitch, which is simply the fundamental frequency of the acoustic wave, notated with an alphabetical letter from A to G. In standard Western equal temperament, the A above middle C, also called A4, has been assigned the fundamental frequency of 440Hz. Interestingly, humans perceive musical intervals as an approximately logarithmic relation to fundamental frequency. The interval between 440Hz and 880Hz sounds identical to the interval between 220Hz and 440Hz. As a result, Western music theory dictates that the frequencies of the standard equally tempered notes are distributed exponentially [20]. Thus, the pitch (frequency) of each note can be found by the following relation,

$$f = f_0 * 2^{n/m}, \tag{1}$$

where f_0 is the reference frequency relative to which one wishes to find a note separated by n semitones and m is the number of equal-tempered semitones in an octave [21]. In the standard Western equal temperament $m = 12$, as there are 12 semitones in each octave. For example, if one desires to find the fundamental frequency of F4, using A4 as the reference, $f_0 = 440$Hz (the fundamental frequency of A4) and $n = -4$ (F4 is four semitones below A4). Using (1), the fundamental frequency of F4 is found to be 349.23Hz.

When it comes to how a note is selected, there are generally two types of instruments, "non-fretted" and "fretted", specifically within the stringed instrument family. Violins are an example of a non-fretted instrument as the musician selects notes by pressing down along the smooth neck. Whereas the violin is capable of playing any note on a continuous spectrum, playing the desired note requires a significant level of skill. On the contrary, the guitar has metal implants called frets that aid in note selection. On a fretted instrument, when the musician presses down near a fret, the effective length of the string is dictated by the metal fret rather than by the musician's finger, resulting in quantization applied to the user input. The concept behind frets is useful for effectively quantizing the musician's input into predefined regions allowing for improved intonation in the presence of imperfect musicianship, which can be viewed as noise. Playing the desired frequency is essential to generating music; however, a technique called "vibrato" produces an audibly pleasing phenomenon by modulating the pitch slightly above and below the center frequency, often heard at the end of vocal phrases.

B. Modern Virtual Instruments

With the rise of personal computers, most modern digital musical instruments are purely virtual. Entire platforms exist for musicians to control numerous parameters, sequence a plethora of sound-bytes, and switch between virtual instruments in real-time. Most virtual instruments rely on the MIDI standard to connect to musical interfaces. As such, a virtual instrument can be easily obtained and loaded into a digital audio workstation (DAW) or live audio platform and can be controlled by a vast number of different interfaces.

The novel Radar Musical Instrument developed in this paper will rely heavily on the music theory as discussed in this

section for note selection, pitch and vibrato generation, and MIDI interface to leverage the features that can be extracted from the mmWave radar return signal. As such, the musical instrument interface detailed in this paper will focus primarily on providing the MIDI signals to a virtual instrument, allowing the virtual instrument to handle the intricacies of the audio output and instrument synthesis. However, a custom real-time audio output tool is also developed along with the musical instrument interface for demonstration and validation. In this manner, the proposed novel Radar Musical Instrument demonstrates the viability of mmWave imaging systems for real-time acute human gesture recognition and computer vision.

III. Radar Theory for Musicians

In this section, we overview the propagation model for the MIMO-FMCW radar chirp signal and examine the spatiotemporal features of a target in motion. The geometry of the Radar Musical Instrument, as shown in Fig. 1, consists of a multistatic linear MIMO array facing vertically. Throughout this paper, the musician's hand is modeled as a point reflector located at the point (y, z). Given the range and cross-range resolution limitations of the radar sensor, dictated by the chirp bandwidth, beam pattern, element locations, and the number of virtual elements, approximating the human hand as a point reflector holds valid for most hand sizes.

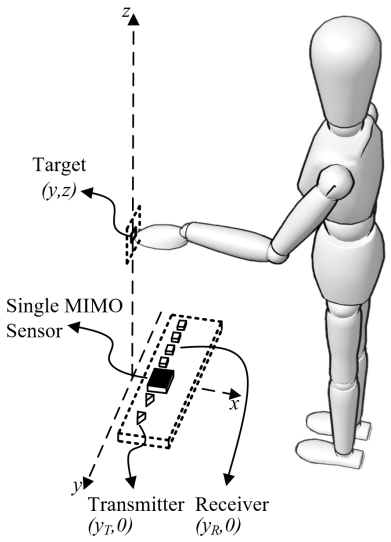

Fig. 1. The geometry of the Radar Musical Instrument, where the linear MIMO array faces vertically and the musician moves their hand throughout the y-z plane.

A. MIMO-FMCW Signal Model

The FMCW chirp signal model is well documented in literature [22] and is described here for reference and continuity throughout this paper. Considering a single transceiver pair located at $(y_T, 0)$ and $(y_R, 0)$in the y-z plane, respectively, the FMCW IF signal or beat signal can be expressed as [23]

$$s(y_T, y_R, k) = \frac{p}{R_T R_R} e^{jk(R_T + R_R)}, \tag{2}$$

where R_T is the distance from the transmitter to the point target, R_R is the distance from the receiver to the point target, $k = \frac{2\pi f}{c}$ is the instantaneous wavenumber of the frequency $f = f_0 + Kt$, f_0 is the carrier frequency at time $t = 0$, K is the chirp slope, and T is the chirp duration in seconds.

(2) shows the multistatic beat signal of a given target, but the monostatic equivalent is much more convenient to work with in the later signal processing steps. Using (y_0, z_0) as a reference point at the center of the target domain, the received multistatic beat signal can be converted to its corresponding monostatic equivalent using the approximation developed in [24] as

$$\hat{s}(y', k) = s(y_T, y_R, k) e^{-jk\frac{d_y^2}{4z_0}}, \tag{3}$$

valid only for small d_y, the distance between the transmitter and receiver elements. 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

$$\hat{s}(y', k) \approx \frac{p}{R^2} e^{j2kR}. \tag{4}$$

From (4), the radial distance from the radar to the target can already be identified as the frequency of the beat signal.

B. FMCW Doppler Radar

Under the monostatic radar regime, whose beat signal is shown in (4), the Doppler effect can be observed by transmitting several chirps at a known pulse repetition interval (PRI), T_{PRI} and tracking the phase change. Now, we extend the model from a stationary point reflector to a target in constant radial velocity. As discussed in detail in [25], the resulting beat signal can be expressed as

$$\hat{s}(y, k, n_c) = \frac{p}{R^2} e^{j(2kR + \frac{4\pi v T_{PRI}}{\lambda_0} n_c)}, \tag{5}$$

where R is the initial range of the target, v is target's constant velocity, λ_0 is the wavelength of the starting frequency of the FMCW chirp f_0, and n_c is the chirp index,

From this equation, it is obvious that the beat signal 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, the traditional method for identifying the target's range and velocity consists of gathering the slow-time series data into a time vs. slow-time data matrix and performing an efficient fast Fourier transform (FFT).

C. Feature Extraction Methods

Now, after the multistatic-to-monostatic conversion in (3), the approximate monostatic signal can be processed to extract spatiotemporal features used to generate music based on the unique characteristics of the hand's position and temporal qualities.

1) Range Migration Algorithm: Taking the monostatic received signal (4), the 2-D position and the velocity of the single point target are now estimated. Whereas traditional methods exist to estimate the range and angle of a target via a 2-D fast Fourier transform (FFT) and simple maximum finding [26], position estimation accuracy is quite low [27]. Instead, we will use a Doppler velocity preserving range migration algorithm (RMA) approach to reconstruct a 2-D range vs. cross-range image of the target scene, from which spatial and temporal information can be extracted.

The primary goal of the RMA is to reconstruct the target scene's reflectivity $p(y, z)$ function. Neglecting the amplitude terms in (4), the approximated monostatic beat signal is the superposition of the backscattered echo signal at every point in the scene, as modeled by

$$\hat{s}(y', k) = \iint p(y, z) e^{j2kR} dy dz. \tag{6}$$

Inverting this equation and simplifying using the method of stationary phase, the reflectivity function $p(y, z)$ can be estimated efficiently by

$$\hat{p}(y, z) = IFT_{2D}^{(k_y, k_z)} \left[\text{STOLT} \left[FT_{1D}^{(y)} [\hat{s}(y', k)] \right] \right], \tag{7}$$

where STOLT[.] is the Stolt interpolation operation and FT and IFT are the forward and backward Fourier transform operators.

2) Doppler Range Migration Algorithm: As discussed above, Doppler velocity information of a target can be estimated by performing an FFT across the slow-time (chirp) dimension of the radar return data; however, the same Doppler velocity information contained in the phase of the return signal is preserved in the RMA algorithm. Notice (5) contains the same beat signal as (4) with an additional, disjoint, frequency corresponding the target velocity. As a result, equivalent techniques can be employed to extract Doppler velocity from the RMA images.

For the signal model above, N_c chirps are captured and stored in for target velocity estimation as $\hat{s}(y', k, n_c)$, where n_c is the chirp index. Once the RMA is performed for each chirp, the image is cropped to include only the region of interest (ROI), wherein we expect to find the hand, and whose dimensions are N_y x N_z. The stored data matrix contains the complex RMA image for each chirp as $\hat{p}(y, z, n_c)$. For the sake of computational efficiency, the cross-range location is estimated and used to narrow the data to two dimensions as $\hat{p}(z, n_c)$. Next, the Doppler FFT is taken across the slow-time dimension yielding the range-Doppler map $d(z, n_d)$ [26], where n_d is the Doppler index corresponding to the Doppler velocities between $[-\frac{\lambda_0}{4T_{PRI}}, \frac{\lambda_0}{4T_{PRI}}]$, T_{PRI} is the chirp periodicity or PRI, and $\hat{y}_p$ is the estimated cross-range position of the target, as shown in (8),

$$d(z, n_d) = FT_{1D}^{(n_c)} \left[\hat{p}(y, z, n_c) \Big|_{y=\hat{y}_p} \right]. \tag{8}$$

IV. The Radar Musical Instrument

In this section, we present the novel Radar Musical Instrument, a mmWave imaging system capable of high accuracy hand tracking for human-computer interaction. The success of the Radar Musical Instrument both demonstrates the specific application of mmWave radar spatiotemporal algorithms for audio signal generation and verifies the legitimacy of such imaging systems for a host of human-computer interaction and computer vision applications.

A. Hardware and Software Setup

The hardware employed in the proposed system consists of a Texas Instruments (TI) automotive radar in conjunction with a real-time data capture adapter. In this research, signal processing, visualization, and machine learning are performed in real-time on a personal computer (PC) in MATLAB. Once the algorithms are validated on a PC, they can be implemented onto such embedded devices for application-specific usage; however, this paper serves as a broad demonstration of the viability of mmWave sensors for real-time computer vision and radar perception applications.

The TI radar is a multistatic MIMO FMCW mmWave radar with an operating bandwidth of 4GHz and a center frequency of 79GHz. In this research, we will be utilizing its linear MIMO array consisting of 2 transmit antenna (TX) elements, separated by $2\lambda_c$, and 4 receive antenna (RX) elements, separated by $\lambda_c/2$, resulting in a virtual array of 8 equally spaced virtual elements separated by $\lambda_c/4$, where λ_c is the wavelength corresponding to the center frequency [26]. Before collecting any meaningful data, the calibration methodology discussed in [24] is adopted to mitigate range bias, constant phase errors, and instrumentation delay.

1) Real-Time Data Retrieval and MATLAB Interface: First, a real-time data reading routine is developed to interface with the data-capture card over its UDP interface. The Radar Musical Instrument requires a custom solution capable of a.) receiving the sequential UDP packets, b.) organizing the packets into entire chirp or frames (sequence of chirps), and c.) providing the data to MATLAB in real-time for signal processing and deep learning.

A custom real-time MATLAB graphical user interface (GUI) is written to serve as the single user interface required for any musician to easily use the Radar Musical Instrument. The MATLAB GUI interfaces with the TI mmWave Studio software [28] to set up the radar board and data capture adapter, and then initializes the UDP interface software to properly read in each packet and organize them into frames based on the user-specified radar parameters. While the radar is continuously capturing responses of the target scene, the beat signal is read into the data retrieval software and is delivered to MATLAB. Even though MATLAB does not offer tremendous computational efficiency, it is fast enough to read in frames at a relatively high rate (50Hz-250Hz). The GUI is capable of extracting and visualizing the range, cross-range, Doppler velocity, and cross-range oscillation in real-time. And finally, the GUI enables precise mapping from the dynamic gestures of the musician to audible notes and a MIDI interface. The custom MATLAB GUI functions either as an end-to-end instrument providing the entire interface, signal processing, and audio output functionality or as a MIDI interface enabling

portability and continuity among a multitude of applications and musical venues.

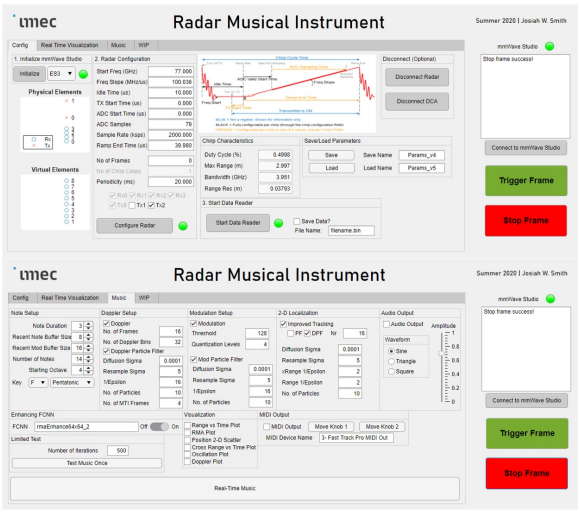

Fig. 2. Radar Musical Instrument custom MATLAB GUI: device setup and music generation pages.

B. Simple Gesture Tracking

This section presents the fundamentals of the Radar Musical Instrument. In its most basic form, the Radar Musical Instrument monitors the spatial and temporal signatures of the musician's hand as it is moved throughout the scene converting these features to music. The three parameters controlling the Radar Musical Instrument audio output are range, cross-range oscillation, and Doppler velocity.

For localization, the RMA is used to reconstruct a 2-D image of the hand in the y-z plane, as discussed in section III-C1. Once the image is reconstructed and cropped to the user-specified region of interest, a simple peak finding routine locates the peak and estimates the 2-D position on a discrete y-z grid. This type of image reconstruction and peak finding suffers from several key factors. First, due to limited bandwidth and the number of antennas, the range and cross-range resolution are on the order of centimeters, even without the presence of noise. Additionally, the RMA assumes an ideal beam pattern from a monostatic full-duplex array, but the physical setup consists of antenna elements with varying, non-ideal near-field beam patterns and a multistatic MIMO array. Whereas the multistatic effects can be somewhat compensated for using the phase correction discussed in section III-A, noticeable image degradation occurs at some positions due to the RMA's monostatic array assumption. These challenges will be further addressed in section IV-C.

Despite the non-ideal imaging environment, the 2-D position of the hand can be estimated within several centimeters. Then the range and cross-range positions are stored in circular buffers. The real-time position estimate is used to control musical output. To select a note, the musician moves their hand vertically through the region of interest. Instead of selecting from a continuous range of frequencies, "virtual frets" are implemented to quantize the range into subregions corresponding to predefined notes. The note selection and virtual fret parameters are all customizable in the Radar Musical Instrument GUI.

Next, as the cross-range location is stored, its oscillation rate is extracted. Due to possible timing issues in the data retrieval tool discussed previously, a non-uniform fast Fourier transform (NUFFT) is performed is used to extract the oscillation rate. Similarly, a NUFFT is implemented to extract the Doppler velocity as discussed in section III-C2.

These temporal features can serve several purposes for musical output. Using the built-in audio output tool, the cross-range oscillation rate is used to control the vibrato specified to each note. If the musician is moving their hand back and forth along the y axis, the Radar Musical Instrument plays the desired note with a vibrato effect at the same rate as the hand. Simply, the frequency of the current note is modulated several Hertz above and below the actual fundamental frequency of the note in a sinusoidal fashion.

Alternatively, the MATLAB GUI is capable of connecting to a virtual instrument via MIDI using the range, cross-range oscillation, and Doppler velocity to control the instrument. In this manner, the Radar Musical Instrument acts as a musical interface controlling the software-based instrument similar to a MIDI keyboard. As with the audio output tool, the MIDI output tool discretizes the range into regions corresponding to MIDI notes (whose possible values range from 0 to 127). Now, the cross-range oscillation rate and Doppler velocity are treated as MIDI controller knobs. The musician can program these features to control any tunable parameter within the virtual instrument enabling novel music generation techniques and dynamic control of a MIDI instrument unique to the Radar Musical Instrument. As a result, precise non-contact gesture control allows artists to generate musical sequences only attainable in real-time using the Radar Musical Instrument.

The entire signal processing chain of the Radar Musical Instrument from the input echo signal to the musical features is shown in Fig. 3. The echo signal is read into MATLAB where the preprocessing discussed in this section is performed and the user inputs are converted into audio or MIDI output by extracting the spatiotemporal features (range, cross-range oscillation, and Doppler velocity) in real-time. In the next section, these three features are called the new noisy measurements or the new noisy measurement vector $\boldsymbol{z}$.

In the simple gesture tracking method, range, cross-range, Doppler velocity, and cross-range oscillation frequency are extracted from the FMCW beat signal using traditional techniques and used to control the musical instrument. The simple tracking techniques discussed in this section allow tracking of spatial and temporal features; however, non-idealities due to noise and RMA assumptions degrade RMA image quality and subsequent spatiotemporal tracking performance. We will next demonstrate an improved tracking technique based on a modified particle filter algorithm and deep learning-aided feature extraction method.

C. Enhanced Gesture Tracking

In this section, we improve the capabilities of the Radar Musical Instrument from the simple tracking techniques for

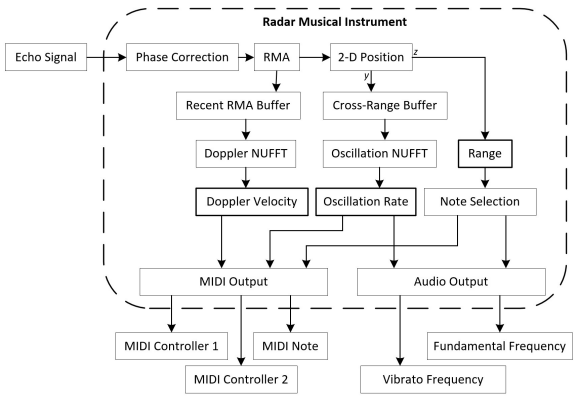

Fig. 3. Radar Musical Instrument signal processing chain: converting the echo signal input to audio or MIDI output.

spatiotemporal tracking and RMA image enhancement to overcome foundational non-idealities in the imaging scenario and inconsistencies in the user input. The concepts demonstrated in this section are applicable novelties for many tracking and high-resolution imaging applications far beyond the scope of HCI.

1) Improved Spatial and Temporal Tracking: We proceed to improve the Radar Musical Instrument's spatiotemporal tracking robustness by utilizing a modified particle filter algorithm [29]. As an example, we begin by employing a particle filter for 2-D spatial localization. A key assumption in our modified model is that the musician's hand is likely to remain stationary, rather than assuming some constant velocity or known deterministic motion model; we will refer to this assumption as the semi-stationary assumption. As a result, our tracking methodology yields highly consistent and smooth localization while finely tracking the hand location throughout the region of interest, but does suffer performance degradation at high hand velocities.

Algorithm 1: Modified Particle Filter Algorithm

randomize n particle states $\boldsymbol{x}_{t-1}$ throughout ROI;
initialize n uniform weights $\boldsymbol{w}_{t-1}$;
while *true* **do**
  retieve beat signal $s(y,k)$;
  extract measurement vector $\boldsymbol{z}$;
  sample particle states $\boldsymbol{x}_{t-1}$, with replacement, using weights $\boldsymbol{w}_{t-1}$ to obtain $\boldsymbol{x}_t$;
  resample particles $\boldsymbol{x}_t = \boldsymbol{x}_t + \boldsymbol{\epsilon}(\boldsymbol{z}_t - \boldsymbol{s}_{t-1}) + \boldsymbol{\psi}$, where $p(\boldsymbol{\psi}) \sim N(\mathbf{0}, \Sigma_\psi)$;
  compute weights $\boldsymbol{w}_t$ by $p(\boldsymbol{z}_t|\boldsymbol{x}_t) \sim N(\boldsymbol{s}_{t-1}, \boldsymbol{\Sigma}_w)$;
  normalize weights $\sum_n \boldsymbol{w}_t^{(n)} = 1$;
  estimate $\boldsymbol{s}_t = \sum_n \boldsymbol{w}_t^{(n)} \boldsymbol{x}_t^{(n)}$;
end

Under the semi-stationary assumption, instead of having a deterministic control input, the effective control input is a weighted movement towards the newest measurement. For 2-D localization, the new noisy measurement, stored in the vector $\boldsymbol{z}$, is made by performing the RMA and locating the peak. $\boldsymbol{z}$ has two elements, the newest estimates of the range, and cross-range. Algorithm 1 details the modified particle filter implementation, using $\boldsymbol{x}$ as the vector containing particle locations in the 2-D plane, $\boldsymbol{w}$ as the vector of weights corresponding to each particle, $\boldsymbol{z}$ as the measurement vector taken at each time step by extracting the features from the beat signal $s(y,k)$, $\boldsymbol{s}$ as the estimated state vector, and $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ as the multivariate Gaussian distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

Proper handling of the key steps, 1) resampling of the particle states and 2) computing new weights, is essential to effectively implementing the semi-stationary modification to the particle filter algorithm.

The particle resampling process involves moving the particles towards the new measurement by a specified weight. $\boldsymbol{\epsilon}$ is a diagonal matrix whose elements weight the importance of each new noisy measurement for the corresponding feature. In this way, the new measurements do not dominate the motion tracking but are included in the localization procedure while bypassing the requirement of a motion model. Fig. 4 demonstrates the resampling process with $\boldsymbol{\epsilon} = \frac{1}{2}\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix. Note that before computing the new weights, particle diffusion is performed by adding the zero-mean Gaussian noise term $\boldsymbol{\psi}$.

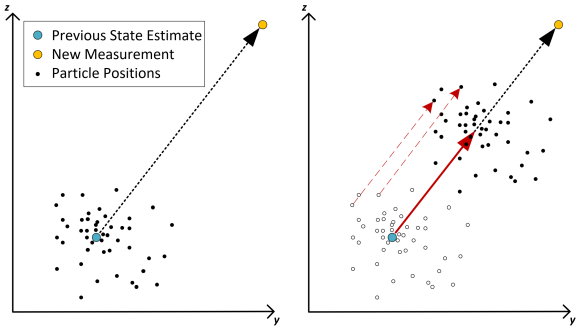

Fig. 4. A visual example of the modified particle filter algorithm resampling process. The particle locations are resampled by a shift transformation towards the new measurement according to the diagonal elements of the weight matrix $\boldsymbol{\epsilon} = \frac{1}{2}\boldsymbol{I}$.

The new weights are computed from the multivariate Gaussian distribution with $\boldsymbol{s}_{t-1}$, the previously estimated states, as the mean vector and a predefined covariance matrix. Therefore, particles closer to the previously estimated state have 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, 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.

Just as the 2-D position can be robustly tracked with our modified particle filter algorithm, an identical implementation can be used to estimate and track the velocity of the target and its cross-range oscillation rate as well, both of which are utilized further in the development of the Radar Musical Instrument.

2) Doppler-Corroborated Real-Time Weighting: The resampling and new weight calculation processes result in a similar technique to the Kalman filter with several advantages. It can easily track non-linear dynamics and does not require prior knowledge of the motion model or noise parameters. Now, we extend the constant weighting method to a real-time weighting technique specific for the range domain. 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 velocity along the range (z) dimension even in the presence of noisy position estimates. First, a Doppler NUFFT is performed across the N_r recent complex RMA images and the Doppler velocity ($\hat{v}_d$) is estimated after video pulse integration by

$$\hat{v}_d = \arg\max \sqrt{\int |\tilde{d}(z, n_d)|^2 dz}, \tag{9}$$

where $\tilde{d}(z, n_d)$ is the recent range-Doppler map as discussed previously. Then, the new measurement is used to calculate the sample velocity ($\hat{v}_s$) based on the recent range locations using the least squares estimator as

$$\hat{v}_s = \frac{N_r \sum_i (\tilde{r}_i t_i) - \sum_i \tilde{r}_i \sum_i t_i}{N_r \sum_i (\tilde{r}_i^2) - \left(\sum_i \tilde{r}_i\right)^2}, \tag{10}$$

$$t_i = 0, T_{PRI}, \ldots, (i-1)T_{PRI}, \ldots, (N_r - 1)T_{PRI}, \tag{11}$$

where $\tilde{r}_i$ is the buffer of recent range estimates. Note that the new range measurement is stored at the end of the recent range estimation buffer, $\tilde{r}_{N_r}$, and all the other elements are the previous estimations by the Doppler-corroborated modified particle filter algorithm.

Now, the difference between the Doppler estimated velocity and sample estimated velocity is computed as $\Delta v = |\hat{v}_d - \hat{v}_r|$ and used in the reward function below to update the weight placed on the new noisy measurement in real-time.

$$\epsilon_r(\Delta v) = \begin{cases} \epsilon_0 \cos\left(\frac{2\pi T_{PRI} \Delta v}{\lambda_0}\right) & \text{if } \Delta v \leq \frac{\lambda_0}{4T_{PRI}} \\ 0 & \text{if } \Delta v > \frac{\lambda_0}{4T_{PRI}} \end{cases} \tag{12}$$

When the sample velocity is close to Doppler velocity, Δv is quite small and the reward function is close to ϵ_0. In this way, the new measurement is corroborated with the more reliable Doppler velocity and weighted accordingly. Outliers and erroneous measurements contradicting the Doppler velocity are given less importance during the particle resampling process.

Now, the modified particle filter algorithm can be extended to include Doppler corroboration of the new measurement. Simply, the range resampling weight ϵ_r is computed by (12) and included in the weighting matrix $\boldsymbol{\epsilon}$.

3) Improved 2-D Position Estimation by Enhancing FCNN: Whereas the Doppler-corroborated modified particle filter algorithm improves the tracking consistency and smoothness, non-idealities such as instrumentation delay, ambient/device noise, multistatic effects, and non-spherical beam patterns remain unaddressed. To solve these issues, we present a fully convolutional neural network (FCNN) for image enhancement that improves the 2-D position estimation and subsequent tracking accuracy. Whereas prior CNN techniques are typically based on far-field assumptions and applied in SAR scenarios with large virtual aperture topologies [30], our enhancement FCNN method operates on near-field images, improves localization even with a small virtual aperture of eight elements, and is trained using a novel technique allowing the network to learn the environment and device noise, near-field beam pattern, and multistatic effects.

The architecture adopted in this paper for the enhancement FCNN is shown in Fig. 5. Four convolution layers of decreasing kernel size are each followed by a nonlinear Rectified Linear Unit (ReLU) layer. Each convolutional layer is zero-padded such that its output is the same size as the input.

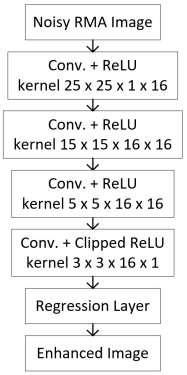

Fig. 5. Architecture of the enhancement FCNN. The selected kernel and layer sizes are capable of adequately learning the non-ideal shape of the distorted RMA image while maintaining high computational efficiency for real-time implementation.

Our enhancement FCNN is trained on both real human hand data and simulated point target data. First, hand data is collected by capturing frames while the user holds their hand at known locations in the ROI. The magnitude of each pixel is taken, $|\hat{p}(y, z)|$, and the real-valued image is the input for training the FCNN allowing the network to fit the non-ideal beam pattern, real multipath and multistatic effects, and actual reflection of a human hand. Next, simulated data is used to supplement the training set. Each simulated sample is generated by simulating a MIMO beat signal with one ideal point target located at a random known location and additive real device noise. The real device noise is captured from the radar with an empty scene. Now, the network learns the device and ambient environment noise as well as the theoretical multistatic effects that remain even after the multistatic-to-monostatic phase correction, thereby improving its generalizability. Our novel training technique results in a robust and generalizable FCNN that improves image signal to noise ratio (SNR) and localization by fitting to the non-ideal imaging constraints.

The output of the FCNN is generated according to the expected images as modeled by

$$\mathcal{I}(y, z) = e^{-(y-y_0)^2/\sigma_y^2 - (z-z_0)^2/\sigma_z^2} \tag{13}$$

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 3dB beamwidth [31]. The output data is generated requiring knowledge of the exact location of the human hand or point target of each input data sample. During training, the FCNN learns to match each noisy, distorted real-valued RMA image to the expected image learning the various non-idealities inherent to the Radar Musical Instrument scenario. Once trained, the network denoises the RMA images and provides a narrow peak at a more precise location allowing for improved localization and subsequent tracking. Results are presented in section V.

Additionally, the enhancement RMA image can be multiplied element-wise by the complex-valued RMA to isolate the complex-valued peak from the hand and mitigate clutter and noise. As a result, the Doppler velocity spectrum SNR is improved, as shown later, since clutter and noise, whose phase adversely affects the Doppler estimation, are greatly reduced. In this way, the enhancement FCNN improves both the spatial and temporal feature estimation in real-time. Pairing the enhancement FCNN with the Doppler-corroborated modified particle filter algorithm, the signal processing chain for the enhanced gesture tracking enabled Radar Musical Instrument is shown in Fig. 6.

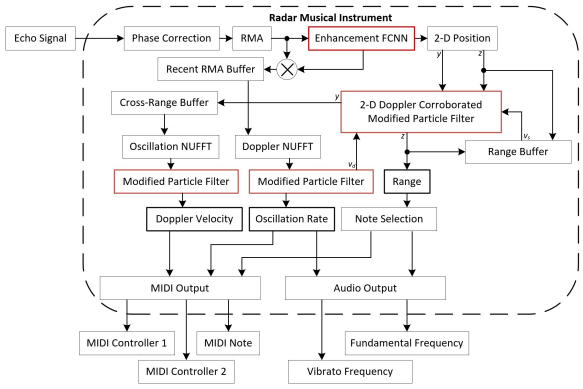

Fig. 6. Radar Musical Instrument signal processing chain from input to musical output now with enhanced gesture tracking by including the Doppler-corroborated modified particle filter algorithm and the enhancement FCNN. Key elements to the enhanced tracking signal processing are highlighted in red.

Utilizing the modified particle filter algorithm and its extension to include Doppler-corroboration improves the robustness of the gesture tracking allowing the Radar Musical Instrument to finely track acute hand gestures even in the presence of noisy user input. Furthermore, the inclusion of the enhancement FCNN enables highly precise position estimation by learning the non-idealities in the device itself and the environment.

V. Results

A. Simple Gesture Tracking Results

As discussed in the previous section, the simple gesture tracking method relies on the traditional techniques presented in section III without any additional tracking algorithm. In real-time, the signal processing chain shown in Fig. 3 is performed, extracting the spatiotemporal features (range, cross-range oscillation, and Doppler velocity) and converting them to musical output via audio or MIDI signals. At each iteration, the states (spatiotemporal features) are estimated directly from the extracted measurements and are therefore prone to erratic behavior.

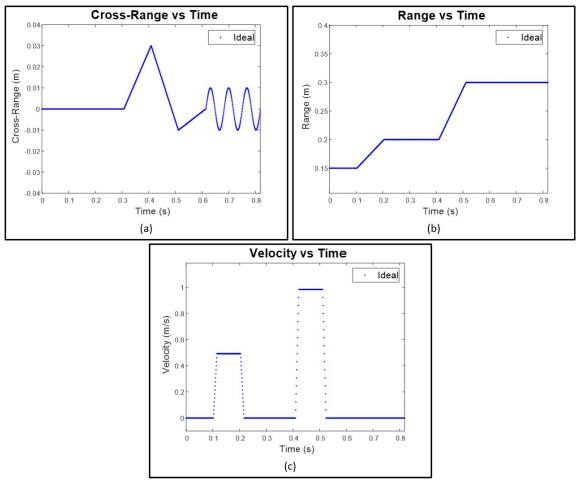

Fig. 7. Ideal motion profile of the target in the (a) cross-range and (b) range directions as well as the (c) range velocity profile against time.

To verify the feature estimation techniques, a virtual prototyping approach is adopted. A point target is simulated in motion with locations and velocity shown in Fig. 7 using (2). Then, real noise collected from the radar with an empty scene is added to each beat signal as

$$\tilde{s}(y_T, y_R, k) = \frac{p}{R_T R_R} e^{jk(R_T+R_R)} + \alpha\tilde{\omega}(y_T, y_R, k), \quad (14)$$

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

The motion profile shown in Fig. 7 shows the range (z), cross-range (y), and velocity 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 from sample to sample. 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.

Fig. 8 shows the features extracted from the simulated noisy beat signals using the simple method. The real radar noise and varying reflectivity result in outliers and errors in the estimated location and velocity of the target. The measurements directly from the RMA image generally follow the motion profile and velocity profile of the target, however suffering from an inherently noisy and inconsistent environment. In the following sections, the performance of the simple gesture tracking approach is quantitatively compared to the enhanced tracking method and design considerations are discussed.[1]

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

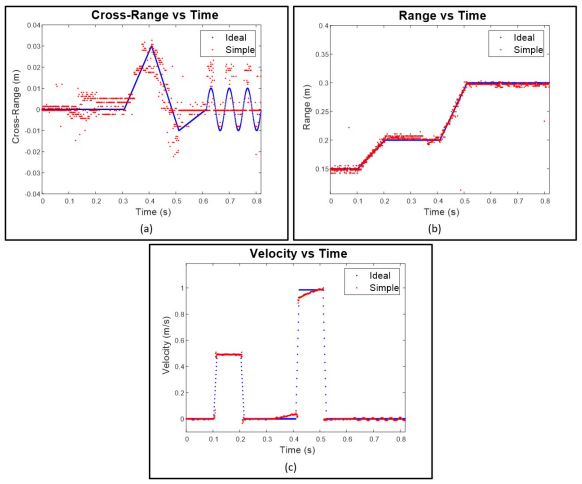

Fig. 8. Motion profile using simple features extraction techniques on each frame for every time step (red) compared with the ideal motion and velocity profiles (blue). The (a) cross-range and (b) range are measured directly from the peak of the RMA image of each frame and the (c) velocity is measured using the Doppler FFT of the raw RMA images using (8) and (9).

B. Enhanced Gesture Tracking Results

Assuming the same motion profile in Fig. 7, the modified 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 musician's control over the Radar Musical Instrument.

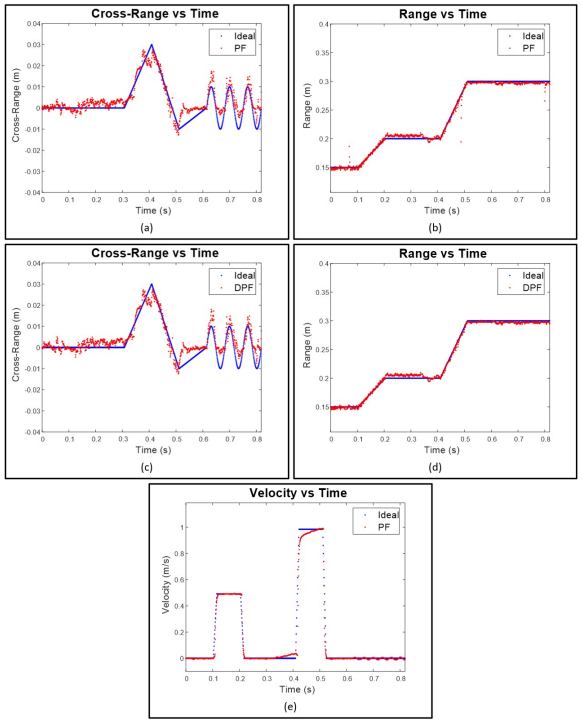

Fig. 9. The modified particle filter (PF) and Doppler-corroborated modified particle filter (DPF) algorithms employed for robust spatiotemporal tracking of the simulated gestures through time: improved tracking of the (a) cross-range and (b) range versus time using the PF, (c) cross-range and (d) range versus time using the DPF with $N_r = 16$, and (e) Doppler velocity versus time using a PF approach.

First, the particle filter algorithm (PF) is implemented using the data in Fig. 8 as elements of the noisy measurement vector $\boldsymbol{z}$. The PF reduces the effect of the noise on the position estimation and improves the spatiotemporal tracking performance as shown in Fig. 9.

Next, the Doppler-corroborated modified particle filter (DPF) is applied to the same set of data additionally improving the state estimation of the range. Notice the outliers in Fig. 9b are mitigated by the DPF in Fig. 9d. This is because the outlying samples result in a sample velocity contradicted by the Doppler velocity and weighted as unimportant. The DPF algorithm improves the playability of the Radar Musical Instrument and the user experience by providing a robust, consistent tracking algorithm to smoothly estimate the 2-D position and spatiotemporal signatures of the musician's gestures. However, key issues discussed previously result in degraded RMA images and subsequent noisy measurements reducing tracking performance.

An image-enhancing fully convolutional neural network is implemented to accommodate non-idealities in the device and environment. 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 $\sigma_y = \sigma_z = 0.001$m resulting in cross-range and range resolutions of 1.18mm. Each simulated sample is generated at a random location in the ROI $z \in [0.1, 0.5], y \in [-0.1, 0.1]$, and the real hand data consists of 512 samples collected at each of the 45 locations throughout the ROI as shown in Fig. 10. The simulated samples cover the entire ROI allowing the network to generalize well to location while learning the non-idealities of the imaging scheme.

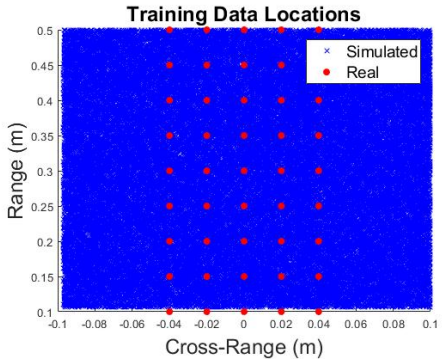

Fig. 10. Locations of the training data used to train the enhancement FCNN. Real data (red) are collected by keeping the hand static at known locations. Simulated data (blue) are generated by choosing locations randomly from the continuous ROI. Simulated data cover nearly the entire ROI allowing the enhancement FCNN to learn locations throughout the ROI.

Training the network for 100 epochs takes 5 hours on a machine with an AMD 3900X processor with 64GB of RAM and 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.

Now, a dataset similar to the training set is collected and simulated for validation. Fig. 11 shows the images enhanced

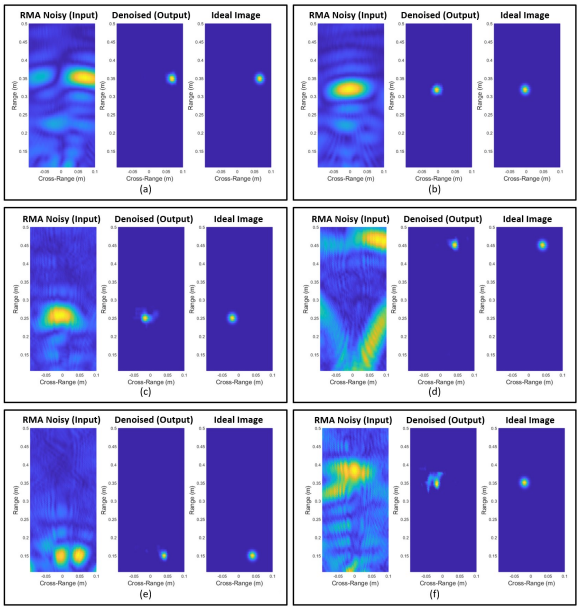

Fig. 11. Enhancement FCNN applied to simulated (a,b) and real hand (c-f) RMA images for image enhancement and improved localization.

by the enhancement FCNN demonstrating the robustness of the network. Figs. 11a and 11b show simulated point targets denoised by the FCNN resulting in improved resolution and more precise localization in the denoised image. Fig. 11c is an RMA image reconstructed 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 capability to enhance images degraded by small hand RCS in comparison to noise, the non-ideal beam pattern of each element resulting in ghosting, ambient and device noise, or other non-idealities.

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 is computed on the validation dataset using the two techniques and shown in Table I. The enhancement FCNN not only improves the resolution of the RMA images but results in more accurate and precise localization for both simulated and real data. Applying the enhancement FCNN can further improve the tracking of the spatiotemporal features.

First, the network improves the Doppler velocity spectrum SNR. Following the signal processing chain in Fig. 6, the real-valued enhanced RMA image is multiplied element-wise with the complex-valued raw RMA image to preserve the target velocity phase term while mitigating noise and clutter. As shown in Fig. 12, the Doppler spectrum SNR is improved when the Doppler processing is performed on the enhanced RMA images as compared with the raw RMA images, reducing the likelihood of erroneously estimating the velocity. As a result, the enhancement network improves the reliability of the Doppler velocity estimation aiding spatial tracking. ‘ Additionally, the enhancement FCNN improves the tracking accuracy of the Doppler-modified particle filter tracking by improving localization accuracy and dependability of the Doppler velocity. The FCNN enhanced Doppler-corroborated modified particle filter algorithm is abbreviated by FCNN-DPF. Fig. 13 demonstrates the tracking using the enhancement FCNN paired with the DPF on the same data as the previous tracking examples. Now, the range and cross-range tracking of the target is nearly identical to the ideal motion profile and an improvement in the velocity estimation.

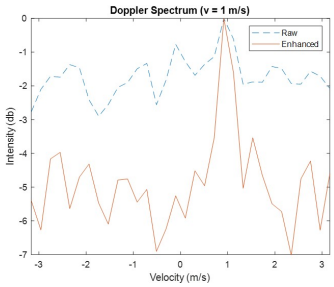

Fig. 12. Comparison of the Doppler velocity spectrum when the Doppler FFT and video pulse integration steps are performed on the raw RMA images compared to the enhanced RMA images. The simulated data contains 128 frames and uses $\alpha = 3$ for every capture to simulate a low SNR scenario.

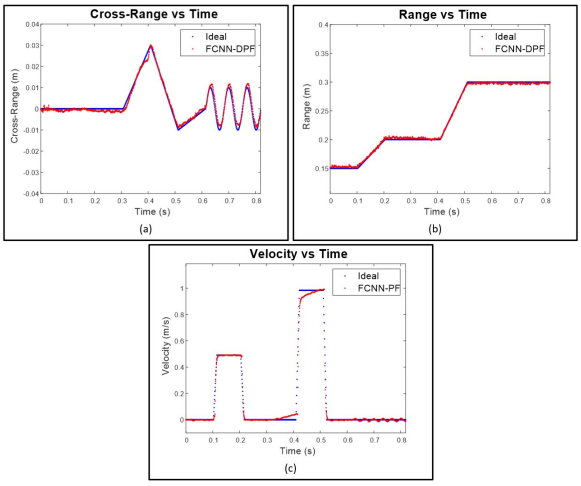

Fig. 13. The FCNN enhanced Doppler-corroborated modified particle filter algorithm.

To quantitatively compare the tracking performance among the various methods described, 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 simple method yields the greatest error for all three features. Comparing PF and DPF, the cross-range and velocity RMSE are identical between the two techniques but the range RMSE is improved due to the real-time range importance weighting. 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 RMSE. The results in Table II demonstrate the considerably superior tracking performance of the enhancement tracking methods, namely the FCNN-DPF, compared with the simple tracking technique along with their respective latencies in the full instrument implementation.

TABLE II
AVERAGE RMSE FOR TRACKING METHODS

	y-Cross-Range (mm)	z-Range (mm)	v-Velocity (m/s)	Latency (ms)
Simple	7.8627	22.0393	0.0724	17
PF	5.2667	13.6412	0.0529	22
DPF	5.2667	6.8543	0.0529	23
FCNN	7.7365	12.2636	0.0584	25
FCNN-PF	3.6974	7.4431	0.0445	28
FCNN-DPF	3.6972	3.0742	0.0445	29

VI. DISCUSSION AND FUTURE WORK

The enhanced gesture tracking method outperforms the simple method in localization accuracy, Doppler SNR, and tracking accuracy; however, there are some necessary trade-offs for this performance gain. First, the effectiveness of the enhancement FCNN is limited by its training set. Specifically, the network is trained on images from a specific region of interest. Due to the limited aperture size, the farther the target is from the radar, the wider the target appears to be. Since the enhancement FCNN is only trained on images within the expected region, extending the ROI outside of the trained region results in performance degradation. In contrast, the simple methods are highly flexible but cannot compete with the performance of the enhancement techniques.

Even though the Doppler-corroborated particle filter improves the tracking robustness, in some cases it can degrade performance. Since the DPF relies on accurate and current Doppler velocity estimation, the system must operate at a fast enough rate that common velocities are within the resolvable range. The UDP interface operates on the order of milliseconds, but the limitation comes in the time between calls to the MATLAB data retrieval function (MEX function). The rate at which the system is acquires the most recent frame is dependent on the speed of the signal processing chain between MEX function calls. As the signal processing time increases, the maximum resolvable velocity can become too small. As a result, real-time tracking performance is degraded since aliasing will occur in the Doppler spectrum domain if the musician's hand is moved too fast. In our work, we found the simple and enhanced methods required similar computation times limiting the frequency of the data retrieval function call to a maximum of around 250Hz. At a starting frequency of 77GHz, this yields a maximum resolvable velocity of 0.24m/s limiting the effectiveness of the DPF when the magnitude of the hand velocity is above this maximum. However, the software package presented in this article is meant to serve as a platform for further development of real-time HCI algorithms on mmWave devices and the latency performance will increase substantially when the algorithms are implemented on an embedded device.

In terms of computational efficiency, the addition of the FCNN and particle filter algorithm increases the computational cost slightly compared to the simple measurement method. However, with GPU acceleration more widely available on most PCs and many MCUs, our comparison shows the latency impact to be negligible among the proposed algorithms. The latencies of the methods introduced in this article show a low quantitative variance. In a small single-blind user study, participants consistently noticed a performance increase as more sophisticated algorithms were introduced, matching the trend in Table II, but were unable to perceive a difference in latency among the algorithms.

The system can be characterized by several key performance metrics. In latency tests, we were able to achieve response times averaging 29ms from gesture movement to MIDI output for the radar musical instrument, comparable to many MIDI interfaces. The latency can be further reduced significantly by implementing the algorithms on an embedded device, but we are pleased with the computational performance on a prototyping platform such as MATLAB. Similarly, the sampling rate of the human hand at around 250Hz can be improved by the same means. Given the scenario and radar under test, the expected range resolution and cross-range resolutions are 3.75cm and 15cm, respectively. However, the proposed methods yield a much improved spatial resolution of 1.96mm and 2.3mm for range and cross-range, respectively. The sensing area of the radar under test is only 0.5m along the range dimension and 0.15m along the cross-range dimension, given the weak reflection from the human hand and small antenna beamwidth. Using a device with more elements or elements with a larger beamwidth would increase the cross-range resolution and sensing area. The software package presented in this article is designed for flexible development and prototyping of algorithms and devices, enabling streamlined hardware changes and upgrades. Lastly, the 2-D position tracking techniques and provided software can easily be extended to 3-D if a 2-D array is employed allowing an added degree of control to the musical instrument at the cost of a larger array topology.

The choice of a mmWave sensor for this application is an effort to prove the concept of acute hand position tracking using radar technology. Compared to optical and RGB+D solutions, mmWave radar is more versatile and reliable operating well under occlusion, in any temperature or lighting environment, and offering 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 tracking applications and serve as a demonstration of their efficacy for HCI and beyond. On the other hand, mmWave sensors cannot meet the performance of optical solutions when it comes to cross-range resolution due to the limited number of radar antennas, 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.

Several alternatives exist to mmWave radar sensing, namely hand-held and optical devices. Hand-held sensing solutions offer highly precise spatiotemporal features, but are not preferable compared to contactless sensors [32], [33]. On the optical front, much effort towards accurate hand position and pose tracking has been made. On the order of $\$100 - \$200$, the Kinect and Leap Motion are in the same price bracket as most mmWave radar devices, which are becoming increasingly inexpensive. Attempts using multiple RGB cameras [34], [35] show promising results; however, a single device is much preferred as multiple cameras setups are cumbersome. Single RGB+D solutions have been proposed using generative pose tracking [36], [37] and learning-based generative pose tracking [38], [39]. However, all of these methods suffer tremendously under occlusion from objects or scene clutter, one of the strengths of mmWave radar. Some deep learning oriented solutions have shown quite promising results [40], [41], but constructing a sufficient dataset for meaningful supervised training remains a challenge.

The Radar Musical Instrument tracks the 2-D position and velocity of the musician's hand to control note selection and two user-selected parameters, a marked improvement over the prior work on mmWave radar tracking only 1-D range for parameter control [16]. However, optical solutions enable tracking of both hands [3], [7], [9], [36]–[41] or hand and finger position [8], [12] for even finer musical control, with some scenario-specific drawbacks. As radar sensor technology improves, tracking the individual fingers on the hand will become increasingly plausible and we expect it to yield comparable or superior results to optical solutions due to higher depth resolution. Compared to prior on gesture tracking with mmWave devices, our proposed methods yield impressive results. Past work using radar devices achieves at best average range tracking error of 2cm [17]. Again using the 4096 simulated motion profiles with added noise, our enhanced gesture tracking technique yields a mean range tracking error of 1.89mm, improving by more than a factor of ten. In [19], a 4GHz bandwidth mmWave sensor is used in conjunction with two optical cameras to achieve a 2-D position RMSE of 1.16mm, at distances closer than 10cm. Comparatively, our enhanced gesture tracking algorithm uses a singular radar device and no optical cameras and still yields a competitive 2-D position RMSE of 3.4mm at a lower cost and computational load. Moreover, our method offers a mean range resolution of 1.96mm, a significant improvement over the 4GHz bandwidth range resolution of 3.75cm. 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 comparison to the Radar Musical Instrument, however, is the Theremin, both being controlled by the hand's proximity to the sensor. The pitch of the Theremin is controlled continuously by the hand's vertical location, whereas the Radar Musical Instrument 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, the Radar Musical Instrument tracks range, cross-range, and velocity, providing three controllable parameters. The Radar Musical Instrument is an evolved Theremin, utilizing a modern mmWave sensor for precise tracking and control of the hand in 2-D space (expansion to 3-D can be easily implemented with the proper hardware), while the Theremin offers a different mode of input, being continuous note selection and using the other hand for volume control at another sensor. 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 available in offline manual transcription and provides the musician a sufficient and consistent level of control and latency, comparable to modern MIDI interfaces. In contrast to a Theremin, the Radar Musical Instrument is significantly less effortful in note selection, allowing simple and intuitive inclusion of the additional parameter controls and increasing accessibility to the expected user-base.

For future work, several promising routes are left to be explored. First, further development of high-throughput radar devices will increase the speed of the real-time data capture process, allowing for increased reliance on the Doppler velocity estimations. Using multiple MIMO radars or a larger MIMO array would enable a multiple-hand and individual finger tracking interface, thus further extending the application space of our robust tracking methods. Additionally, the MATLAB-based system implementation in this paper can be implemented on high-speed DSP controllers to create a portable system. Finally, the novel precise gesture tracking algorithms of Radar Musical Instrument can easily be extended to offer an elegant, efficient solution to a host of acute gesture tracking problems.

VII. Conclusion

The Radar Musical Instrument successfully demonstrates the viability of acute human hand gesture tracking for human-computer interaction using mmWave sensors. We validated and implemented our real-time spatiotemporal signal processing algorithms and robust tracking algorithms in the form of a musical interface; however, the Radar Musical Instrument demonstrates the broad effectiveness of mmWave technology for a multitude of near-field acute gesture tracking applications. First, simple feature extraction and tracking methods were introduced, followed by an enhanced approach leveraging the Doppler-corroborated modified particle filter algorithm and enhancement FCNN to achieve highly robust and accurate tracking in the presence of ambient noise and offering compensation for device non-idealities and multipath effects. The methods are compared demonstrating noticeable improvement using the FCNN-DPF. Additionally, our work offers superior tracking estimation and localization accuracy compared to prior methods in the literature for both mmWave and optical implementations. The novel Radar Musical Instrument presented in this paper offers an elegant solution to a myriad of

contactless human-computer interaction problems far beyond the scope of musical interfaces.

ACKNOWLEDGMENT

This 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 *ICMC*, 1995, p. 26.
[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 *ICMC*, 2011.
[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 *NIME*, 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 *NIME*, 2012.
[6] R. Schramm, C. R. Jung, and E. R. Miranda, "Dynamic time warping for music conducting gestures evaluation," *IEEE Transactions on Multimedia*, vol. 17, no. 2, pp. 243–255, 2015.
[7] A. R. Jensenius, "Kinectofon: Performing with shapes in planes," 2013.
[8] J. Han and N. Gold, "Lessons learned in exploring the leap motion™ sensor for gesture-based instrument design." Goldsmiths University of London, 2014.
[9] L. Hantrakul and K. Kaczmarek, "Implementations of the leap motion in sound synthesis, effects modulation and assistive performance tools." in *ICMC*, 2014.
[10] D. Brown, N. Renney, A. Stark, C. Nash, and T. Mitchell, "Leimu: Gloveless music interaction using a wrist mounted leap motion," 2016.
[11] A. Tindale, A. Kapur, and G. Tzanetakis, "Training surrogate sensors in musical gesture acquisition systems," *IEEE Transactions 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. of CMMR*, vol. 13, 2013.
[13] M. Akbari and H. Cheng, "Real-time piano music transcription based on computer vision," *IEEE Transactions 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*, pp. 1–1, 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 *NIME*, 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 *12th USENIX Symposium on Networked Systems Design and Implementation NSDI 15)*, 2015, pp. 189–204.
[18] Y. Sun, X. Liang, H. Fan, M. Imran, and H. Heidari, "Visual hand tracking on depth image using 2-d matched filter," in *2019 UK/ China Emerging Technologies (UCET)*, August 21-22, 2019, Glasgow, United Kingdom, pp. 1–4.
[19] 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)*, January 4-6, 2020, Las Vegas, NV, USA,, pp. 1–9.
[20] H. Helmholtz, *On the sensations of tone*. Courier Corporation, 2013.
[21] C. Schmidt-Jones, "Understanding basic music theory," 2013.
[22] J. W. Smith, M. E. Yanik, and M. Torlak, "Near-field mimo-isar millimeter-wave imaging," in *2020 IEEE Radar Conference (RadarConf20)*, 2020, pp. 1–6.
[23] 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.
[24] 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.
[25] V. Winkler, "Range doppler detection for automotive fmcw radars," in *Proc. European Radar Conf.*, October 10-12, 2007, Munich, Germany, pp. 166–169.
[26] S. Rao, "Introduction to mmwave sensing: Fmcw radars."
[27] J. Kim, J. Chun, and S. Song, "Joint range and angle estimation for fmcw mimo radar and its application," *arXiv preprint arXiv:1811.06715*, 2018.
[28] "Texas instruments mmwave studio." [Online]. Available: https://www.ti.com/tool/MMWAVE-STUDIO
[29] J. García, A. Gardel, I. Bravo, J. L. Lázaro, and M. Martínez, "Tracking people motion based on extended condensation algorithm," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 43, no. 3, pp. 606–618, 2013.
[30] 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.
[31] 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.
[32] L. Pardue and W. Sebastian, "Hand-controller for combined tactile control and motion tracking." in *NIME*, 2013, pp. 90–93.
[33] 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: An International Journal*, 2010.
[34] L. Ballan, A. Taneja, J. Gall, L. Van Gool, and M. Pollefeys, "Motion capture of hands in action using discriminative salient points," in *European Conference on Computer Vision*. Springer, 2012, pp. 640–653.
[35] S. Sridhar, A. Oulasvirta, and C. Theobalt, "Interactive markerless articulated hand motion tracking using rgb and depth data," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 2456–2463.
[36] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Efficient model-based 3d tracking of hand articulations using kinect." in *BmVC*, vol. 1, no. 2, 2011, p. 3.
[37] 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 *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3325–3333.
[38] S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt, "Fast and robust hand tracking using detection-guided optimization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3213–3221.
[39] 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 Transactions on Graphics (TOG)*, vol. 35, no. 4, pp. 1–12, 2016.
[40] J. Tompson, M. Stein, Y. Lecun, and K. Perlin, "Real-time continuous pose recovery of human hands using convolutional networks," *ACM Transactions on Graphics (ToG)*, vol. 33, no. 5, pp. 1–10, 2014.
[41] Q. Ye, S. Yuan, and T.-K. Kim, "Spatial attention deep net with partial pso for hierarchical hybrid hand pose estimation," in *European conference on computer vision*. Springer, 2016, pp. 346–361.