A Deep Iterative Refinement Receiver for OTFS Symbol Detection in Doubly-Dispersive Channels

Efe Ispir and Ian P. Roberts

***Abstract*—Orthogonal time frequency space (OTFS) modulation has emerged as a promising candidate for high-mobility wireless communication systems due to the diversity it offers across both time and frequency. Reliable OTFS detection, however, remains challenging under doubly-dispersive channels, where delay and Doppler dispersion induce structured interference between transmitted symbols and complicate symbol recovery. To address these challenges, we propose a two-stage iterative OTFS detector that integrates a physics-informed learned initializer with an iterative refinement network, enabling progressively more accurate symbol estimates in doubly-dispersive channels. The initializer incorporates the known delay–Doppler input–output relationship to produce a robust first-stage estimate, while the refinement stage iteratively suppresses residual symbol interference in the delay–Doppler domain. Simulation results demonstrate that the proposed detector achieves consistent performance gains over conventional and existing learning-based detectors across a variety of channel conditions. These results highlight the effectiveness of incorporating known channel structure into the detection process and using iterative refinement for improved and robust OTFS detection.**

***Index Terms*—Orthogonal time frequency space modulation, doubly-dispersive channels, symbol detection, deep learning, high-mobility wireless communications.**

I. Introduction

High-mobility wireless communication scenarios, such as vehicle-to-vehicle networks, low Earth orbit satellite systems, and high-speed railway communications, pose significant challenges for conventional modulation schemes [1], [2]. The high mobility in these environments introduces doubly-dispersive channels, exhibiting both delay and Doppler spreading [1]. Under such conditions, the performance of conventional orthogonal frequency-division multiplexing (OFDM) systems degrades significantly, as high Doppler spread introduces inter-carrier interference (ICI) by breaking subcarrier orthogonality [3], [4].

In such scenarios, orthogonal time frequency space (OTFS) modulation [5] has emerged as a promising alternative to OFDM. By multiplexing symbols in the delay–Doppler domain, OTFS spreads each symbol across both time and frequency, enabling the system to exploit the full diversity of the wireless channel and allowing it to outperform OFDM in doubly-dispersive channels [5]–[8]. Exploiting these advantages requires reliable symbol detection, however, which becomes increasingly difficult as spreading worsens in both delay and Doppler.

The authors are with the Department of Electrical and Computer Engineering, University of California, Los Angeles (UCLA), Los Angeles, CA USA. Email: {efeispir, ianroberts}@ucla.edu.

Although maximum a posteriori (MAP) detection is optimal in theory, under uniform symbol priors it is equivalent to maximum likelihood (ML) detection, which requires an exhaustive search over all possible transmitted symbol vectors [9]. As the OTFS frame size increases, the complexity of this search grows exponentially, making MAP/ML detection impractical for real-time implementation [9]. To address this, the authors of [7] proposed a low-complexity message passing (MP) algorithm. The main shortcoming of this approach arises when multiple transmitted symbols are coupled through more than one shared observation. This introduces cycles into the factor graph, which can prevent belief propagation from converging and, even when it does converge, can lead to a suboptimal fixed point [10], [11]. This limitation becomes more pronounced under fractional Doppler shifts, where energy leakage across Doppler bins increases the coupling among transmitted symbols, resulting in a more densely connected factor graph. To address these convergence issues, [12] proposed a variational Bayes (VB) detector, which approximates the posterior distribution via variational inference, leading to improved convergence behavior. However, they rely on the mean field approximation, which neglects posterior correlations among symbols by assuming a factorized distribution, potentially resulting in underestimated posterior variance [12].

Another conventional approach to OTFS symbol detection is the use of linear equalizers, namely least squares (LS) and minimum mean square error (MMSE) equalizers [9]. While these methods provide effective symbol detection performance, they require matrix inversions whose computational complexity becomes significant for practical OTFS frame sizes. To address this, several low-complexity detectors have been proposed that achieve performance comparable to MMSE detection while reducing computational cost [13]–[15].

Although these model-based detectors achieve good performance, they rely on various assumptions, approximations, and simplifications regarding the channel and interference structure [7], [9], [12]. In contrast, data-driven approaches have proven capable of learning these complex input–output relationships directly from data, offering improved robustness to such modeling inaccuracies. Among these approaches, convolutional neural networks (CNNs) have proven particularly well-suited to the delay–Doppler representation of OTFS signals thanks to their capable processing of two-dimensional data [16]–[18]. In [19], for instance, the authors proposed a CNN-based detector that uses MP estimates as auxiliary input features concatenated with the OTFS received signal to enhance detection performance. Several subsequent works follow a similar strategy by augmenting the CNN input with

initial estimates obtained from conventional methods such as MMSE, LS, or MP [20]–[23]. In [21], standard convolution layers are replaced with depthwise separable convolutions to reduce computational complexity while maintaining comparable detection performance. In [22], wavelet decomposition of the received OTFS frame is combined with the MP estimates to enrich the CNN input representation and improve detection performance. Another related work [23] incorporates MMSE estimates as prior information into a detector based on a residual channel attention network (RCAN) [24] and evaluates its robustness to hardware impairments. Since all these methods rely on auxiliary estimates generated by conventional detectors such as LS, MMSE, or MP, their performance is influenced by the quality of the initial estimate. Moreover, in each of these approaches, the CNN performs a single inference pass, meaning its detection performance is constrained by how effectively it can exploit the information provided by the initial estimate and the received signal within that single pass.

Another approach used in CNN-based detection is to extract a channel embedding and concatenate it with the input. In [25], the embedding is derived from the channel estimates and concatenated with a preprocessed input obtained via a pad-and-slice operation. In [26], the channel embedding is extracted directly from pilot observations. In both cases, channel information is encoded into a learned embedding and concatenated with the input, leaving the network to implicitly learn how this additional information should inform symbol detection. Prior work has also investigated the performance of different CNN-based architectures for OTFS symbol detection [27], but these architectures rely solely on the received OTFS signal, without explicitly incorporating channel knowledge. This places additional burden on the CNN to implicitly learn channel-induced distortions and approximate the inverse mapping, thereby limiting performance.

While CNN-based receivers have shown promising performance, existing approaches present two limitations. First, when channel information is incorporated, it is typically provided through a learned channel embedding concatenated with the input, rather than being explicitly incorporated into the feature extraction process. Thus, the role of the available channel knowledge in the detection process is learned implicitly by the network. Second, existing approaches generally operate in a single-shot inference manner, either directly estimating transmitted symbols from the received signal or adopting hybrid architectures that incorporate prior symbol estimates from conventional detectors as auxiliary information for symbol detection. Given the delay–Doppler-domain interference induced by doubly-dispersive channels, accurately resolving these interference effects within a single inference pass places a substantial burden on the detector.

A. Contributions

To address these limitations of prior work, we propose a two-stage CNN-based detection framework that combines a channel-aware initialization stage with an iterative refinement stage. The initialization stage explicitly incorporates channel information during feature extraction to improve the robustness of the initial estimate, and the refinement stage then progressively updates a detection state through iterative inference using feedback from previous estimates and the received signal. Together, these components enable more reliable detection under doubly-dispersive channels. The principal contributions of this paper can be summarized as follows:

- We propose a two-stage CNN-based detector for OTFS signal detection, where the first stage performs channel-conditioned detection through a delay–Doppler alignment transform and channel-conditioned kernels. By explicitly incorporating channel structure into the feature extraction process, this stage produces an initial symbol estimate that generalizes reliably to unseen channel realizations, providing a robust starting point for the subsequent refinement stage.
- We introduce an iterative refinement architecture in which a learned detection state is progressively updated via gated iterative updates. Symbol probability estimates from previous iterations, along with reconstructed signals and the received signal, are used as feedback to guide interference suppression, while a per-iteration gating mechanism adaptively controls the contribution of new updates, enabling progressive self-correction under doubly-dispersive channel effects.

Experimental results across a wide variety of channel conditions show that the proposed method achieves lower bit error rate (BER) than conventional methods like MMSE and MP, while also outperforming single-shot CNN-based detectors. Furthermore, we observe that the channel-conditioning mechanism substantially improves the generalization of the initializer to unseen channel realizations, while the iterative refinement stage progressively reduces residual detection errors, leading to a lower error floor and more reliable detection in high-mobility scenarios. Finally, we compare the inference time of the proposed method against relevant baselines under both single-frame and batched inference settings.

B. Notation

Bold lowercase letters denote vectors $\mathbf{x}$ or three-dimensional probability tensors $\hat{\mathbf{p}}$, while bold uppercase letters $\mathbf{X}$ denote matrices and higher-order tensors. Non-bold indexed variables, such as $X[\ell, k]$ or $x[q]$, represent individual elements of their corresponding quantities. The operators $(\cdot)^*$, $\mathbb{E}[\cdot]$, $[\cdot]_N$, $\Re\{\cdot\}$, and $\Im\{\cdot\}$ denote the conjugate, expectation, modulo N, real-part, and imaginary-part operations, respectively. $\mathbb{R}$ and $\mathbb{C}$ denote the sets of real and complex numbers, respectively, and $\mathcal{CN}(\mu, \sigma^2)$ represents the circularly symmetric complex Gaussian distribution with mean μ and variance σ^2. The notation $\delta(\cdot)$ denotes the Dirac delta function.

II. System Model

In this section, we describe the considered point-to-point single-input single-output OTFS communication system employing rectangular transmit and receive pulses, including its transmit signal model, doubly-dispersive channel model, and received signal model.

A. OTFS Transmit Signal Model

We consider OTFS frames consisting of N time slots and M subcarriers. The duration of a single time slot is given by $T = T_{\mathrm{s}}M$ where T_{s} denotes the sampling period. The subcarrier spacing is given by $\Delta f = \frac{1}{T}$. Consequently, the overall frame duration and total bandwidth are $T_{\mathrm{f}} = TN$ and $B = \Delta f M$, respectively.

OTFS modulation is formulated on a discretized delay–Doppler plane, with delay resolution

$$\Delta\tau = \frac{1}{M\Delta f}, \tag{1}$$

and Doppler resolution

$$\Delta\nu = \frac{1}{NT}. \tag{2}$$

The resulting grid comprises $M \times N$ discrete locations, indexed by $\ell = 0, \ldots, M-1$ and $k = 0, \ldots, N-1$. For each OTFS frame, MN information symbols, drawn from the alphabet $\mathcal{A}$ of size Q, are arranged in this grid and denoted by $X_{\mathsf{DD}}[\ell, k]$. The transmitter then maps these symbols into the time–frequency domain via the inverse symplectic fast Fourier transform (ISFFT) as

$$X_{\mathsf{TF}}[n, m] = \frac{1}{\sqrt{MN}} \sum_{\ell=0}^{M-1} \sum_{k=0}^{N-1} X_{\mathsf{DD}}[\ell, k] e^{\mathrm{j}2\pi\left(\frac{nk}{N} - \frac{m\ell}{M}\right)}, \tag{3}$$

where $m = 0, \ldots, M-1$ and $n = 0, \ldots, N-1$.

Then, a Heisenberg transform, parametrized by the transmit pulse $g_{\mathrm{tx}}(t)$, is applied to the time–frequency domain samples to obtain the continuous-time transmit signal

$$s(t) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} X_{\mathsf{TF}}[n, m] g_{\mathrm{tx}}(t - nT) e^{\mathrm{j}2\pi m\Delta f(t-nT)}. \tag{4}$$

B. Doubly-Dispersive Channel

The doubly-dispersive wireless channel can be represented in the delay–Doppler domain as

$$h(\tau, \nu) = \sum_{i=0}^{P-1} h_i \delta(\tau - \tau_i)\delta(\nu - \nu_i), \tag{5}$$

where P denotes the number of propagation paths in the channel, while h_i, τ_i, and ν_i represent the complex channel gain, delay, and Doppler shift associated with the i-th propagation path, respectively. The corresponding normalized delay and Doppler parameters are given by

$$\ell_i \triangleq \frac{\tau_i}{\Delta\tau}, \qquad k_i + \kappa_i \triangleq \frac{\nu_i}{\Delta\nu}, \tag{6}$$

where ℓ_i and k_i represent the integer delay and Doppler indices, respectively, with $-0.5 < \kappa_i \leq 0.5$ denoting the fractional Doppler component.

Using the delay–Doppler channel representation, the received signal is given by

$$r(t) = \iint h(\tau, \nu) s(t - \tau) e^{\mathrm{j}2\pi\nu(t-\tau)} \mathrm{d}\tau \mathrm{d}\nu \; + \; w(t), \tag{7}$$

where $w(t)$ denotes additive noise.

C. OTFS Received Signal Model

At the receiver, following standard practice, the time–frequency representation is obtained via the cross-ambiguity function between the matched filter and the received signal

$$Y_{\mathsf{TF}}(t, f) = \int g_{\mathrm{rx}}^{*}(t' - t) r(t') e^{-\mathrm{j}2\pi f(t'-t)} \mathrm{d}t'. \tag{8}$$

The continuous time–frequency representation is then sampled as

$$Y_{\mathsf{TF}}[n, m] = Y_{\mathsf{TF}}(t, f)\big|_{t=nT, f=m\Delta f}. \tag{9}$$

The signal can then be transformed back to the delay–Doppler domain by applying the symplectic fast Fourier transform (SFFT) to the sampled time–frequency representation, resulting in

$$Y_{\mathsf{DD}}[\ell, k] = \frac{1}{\sqrt{MN}} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} Y_{\mathsf{TF}}[n, m] e^{-\mathrm{j}2\pi\left(\frac{nk}{N} - \frac{m\ell}{M}\right)}. \tag{10}$$

Under fractional Doppler shifts, the above operation results in the spreading of each transmitted symbol along the Doppler dimension, causing inter-Doppler interference (IDI) [7]. As shown in [7], this Doppler-domain spreading can be approximated by retaining only the N_i dominant leakage components on each side of the peak component, where $N_i \ll N$. The resulting approximation is given by [7]

$$\begin{aligned} Y_{\mathsf{DD}}[\ell, k] \approx \sum_{i=0}^{P-1} \sum_{q=-N_i}^{N_i} & h_i e^{\mathrm{j}2\pi(k_i+\kappa_i)\frac{\ell-\ell_i}{MN}} \alpha_i(\ell, k, q) \\ & \times X_{\mathsf{DD}}[[\ell - \ell_i]_M, \; [k - k_i + q]_N], \end{aligned} \tag{11}$$

where

$$\alpha_i(\ell, k, q) = \begin{cases} \frac{1}{N}\beta_i(q) & \ell_i \leq \ell < M \\ \frac{1}{N}(\beta_i(q) - 1) e^{-\mathrm{j}2\pi\frac{[k-k_i+q]_N}{N}} & 0 \leq \ell < \ell_i, \end{cases} \tag{12}$$

and

$$\beta_i(q) = \frac{e^{-\mathrm{j}2\pi(-q-\kappa_i)} - 1}{e^{-\mathrm{j}\frac{2\pi}{N}(-q-\kappa_i)} - 1}. \tag{13}$$

III. Proposed Two-Stage Framework

As shown in [28], communication over doubly-dispersive channels introduces both delay and Doppler dispersion, resulting in time–frequency selectivity and structured interference in the delay–Doppler domain. While CNN-based receivers have demonstrated promising performance in symbol detection, they typically operate as *single-shot* detectors. These approaches either directly estimate transmitted symbols $X_{\mathsf{DD}}[\ell, k]$ from the received signal $Y_{\mathsf{DD}}[\ell, k]$ [25]–[27] or employ hybrid architectures that use prior symbol estimates generated by conventional detectors and apply a single neural network-based refinement step [19]–[23]. In this work, we refer to the latter category as *hybrid single-shot* detectors. Although hybrid single-shot detectors may use channel information in their conventional initialization stage, the neural component typically receives only the resulting

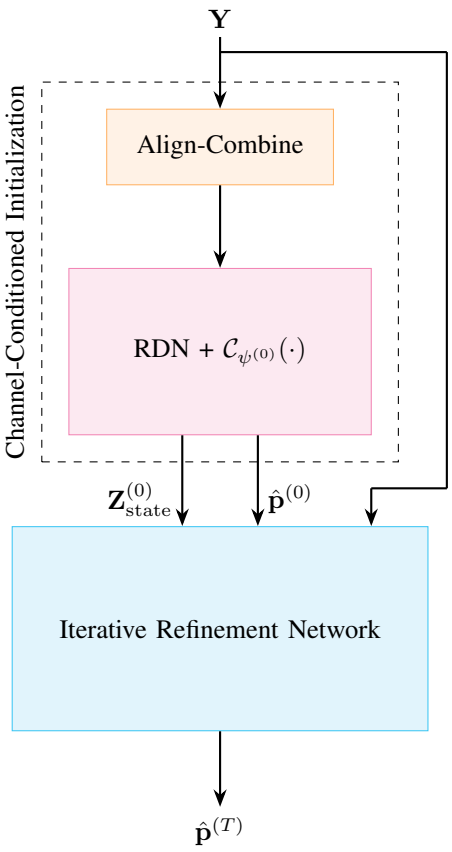

Fig. 1. Overview of the proposed iterative OTFS detection framework. The received signal $\mathbf{Y}$ is first processed by a channel-conditioned initialization module consisting of an Align-Combine operation, an RDN, and a classifier head. This produces the initial symbol probability estimate $\hat{\mathbf{p}}^{(0)}$ and detection state $\mathbf{Z}^{(0)}_{\text{state}}$, which are then iteratively refined by the proposed iterative refinement network (IRN). The final output $\hat{\mathbf{p}}^{(T)}$ is obtained after T refinement iterations.

symbol estimates rather than the channel information itself. In contrast, other approaches explicitly incorporate channel information by providing learned channel embeddings [25], [26]. Alternatively, some methods rely solely on the received signal without explicit channel information [27].

This motivates two complementary design choices in our receiver. First, rather than treating channel information solely as an additional learned representation, we incorporate the known delay–Doppler input–output structure directly into the feature extraction process. Second, rather than producing a single-shot estimate, we iteratively refine the detection result to progressively correct residual errors. These choices lead to a two-stage OTFS symbol detection framework (see Fig. 1), consisting of a channel-conditioned initialization stage followed by an iterative refinement stage. The channel-conditioned initialization stage can be viewed as a single-shot detector that incorporates the proposed Align-Combine module to explicitly exploit channel information during the initial symbol estimation process. The iterative refinement stage then progressively refines the initial symbol estimates through the proposed iterative refinement network (IRN). The effectiveness of the proposed detection framework is later evaluated through extensive simulations under various channel conditions, demonstrating its ability to improve upon both conventional and single-shot CNN-based detectors.

A. Channel-Conditioned Initialization

The channel-conditioned initialization module generates the initial symbol estimates and the detection state that serve as the starting point for the subsequent IRN. It consists of the proposed Align-Combine module, which constructs a channel-conditioned feature representation, followed by a residual dense network (RDN) [29] that extracts and refines the channel-conditioned features to produce the initial detection state, and a classifier head that maps this state to initial symbol probability estimates. The RDN is adopted as the CNN backbone based on prior OTFS detection results showing that its dense connectivity and effective feature reuse provide favorable detection performance compared with other CNN architectures [27]. As we will show in Section IV, removing the Align-Combine module leads to a significant degradation in generalization to unseen channel conditions, highlighting the importance of explicit channel conditioning in the feature extraction process. This channel conditioning is motivated by the structured input–output relationship that OTFS exhibits in the delay–Doppler domain, as characterized by (11), where each propagation path induces a specific delay–Doppler shift of the transmitted symbols. Align-Combine therefore organizes and combines the received signal according to these shift relationships and the corresponding channel information, providing the subsequent detection network with a channel-aware feature representation rather than requiring these relationships to be learned solely from data.

The Align-Combine module is composed of two distinct steps. The first step applies a predetermined set of circular shifts to the received delay–Doppler frame to construct aligned copies corresponding to all possible integer delay–Doppler displacements. This operation assumes knowledge of the channel's maximum (worst-case) delay and Doppler support. The resulting circular shifts correspond to the inverse shifts of the delay–Doppler displacements in (11). The c-th circularly shifted copy is defined as

$$Y^{(c)}_{\text{align}}[\ell, k] = Y_{\text{DD}}[[\ell + \ell^{(c)}]_M,\ [k + k^{(c)}]_N], \qquad (14)$$

where $\ell^{(c)} \in \{0, \dots, \ell_{\max}\}$, $k^{(c)} \in \{-k_{\max}, \dots, k_{\max}\}$, and $c = 0, \dots, C-1$ with

$$C = (\ell_{\max} + 1)(2k_{\max} + 1), \qquad (15)$$

representing the total number of integer delay–Doppler channel taps supported by the channel. The superscript (c) denotes the delay–Doppler shift associated with the c-th channel tap and distinguishes these shifts from the path-dependent shifts ℓ_i, k_i in (11).

These aligned copies are then concatenated along the feature channel dimension to form a real-valued tensor, where the real and imaginary components are treated as separate feature

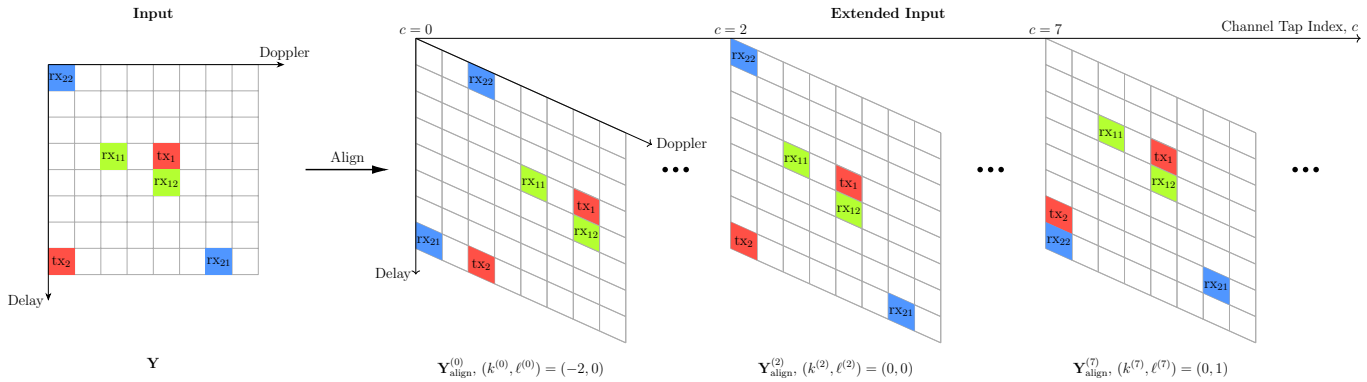

Fig. 2. Illustration of the alignment stage of the proposed Align-Combine function. For ease of visualization, a simplified channel model with integer Doppler shifts and no Doppler spread is considered, with maximum supported Doppler and delay shifts $k_{\max} = 2$ and $\ell_{\max} = 1$, respectively, and two multipath components having Doppler and delay shifts $(k_1, \ell_1) = (-2, 0)$ and $(k_2, \ell_2) = (0, 1)$. The transmitted OTFS frame contains only two symbols, tx_1 and tx_2, shown in red. The green and blue cells represent the corresponding received copies, denoted by rx_{ij}, where i indexes the transmitted symbol and j indexes the multipath component. The left-hand side shows the received delay–Doppler domain input to the alignment stage. The right-hand side illustrates the resulting complex extended input, where the axis c indexes the channel taps characterized by Doppler and delay shifts $(k^{(c)}, \ell^{(c)})$, with each grid representing the corresponding circularly shifted input defined in (14). Only a subset of the aligned copies is shown for ease of visualization; the complete extended input contains all $C = 10$ supported delay–Doppler shifts (see (15)).

channels. This yields a real-valued tensor $\mathbf{Y}_{\mathrm{ext}} \in \mathbb{R}^{M \times N \times 2C}$ of the form

$$\mathbf{Y}_{\mathrm{ext}} = \Big[\Re(\mathbf{Y}^{(0)}_{\mathrm{align}}),\, \Im(\mathbf{Y}^{(0)}_{\mathrm{align}}),\, \Re(\mathbf{Y}^{(1)}_{\mathrm{align}}),\, \Im(\mathbf{Y}^{(1)}_{\mathrm{align}}),\, \ldots,\, \Re(\mathbf{Y}^{(C-1)}_{\mathrm{align}}),\, \Im(\mathbf{Y}^{(C-1)}_{\mathrm{align}})\Big]. \tag{16}$$

Fig. 2 provides a visual interpretation of this construction, where each grid along the c-axis represents the received frame after a different circular shift associated with an integer delay–Doppler displacement. The resulting stack therefore organizes the received signal according to the shift structure induced by the supported channel taps.

The second step of Align-Combine applies a channel-conditioned convolution to the extended feature tensor $\mathbf{Y}_{\mathrm{ext}}$. We extract different kernels for the integer and fractional Doppler cases. For integer Doppler shifts, we employ kernels with spatial size (1×1) as the channel response corresponds to an exact shift without Doppler-domain energy spreading. In contrast, for fractional Doppler shifts, where Doppler leakage spreads energy across neighboring bins, we use kernels with spatial size $(1 \times 2N_i + 1)$ to capture the dominant Doppler-domain leakage components.

1) Integer Doppler Kernel Generation: For the integer Doppler case, we first construct a sparse estimated channel representation given by

$$\tilde{h}_c = \begin{cases} \sum_{i \in \mathcal{P}_c} \hat{h}_i, & \mathcal{P}_c \neq \emptyset \\ 0, & \text{otherwise} \end{cases}$$
$$\mathbf{h}_{\mathrm{in}} = [\Re(\tilde{h}_0), \Im(\tilde{h}_0), \ldots, \Re(\tilde{h}_{C-1}), \Im(\tilde{h}_{C-1})] \in \mathbb{R}^{2C}, \tag{17}$$

where $\mathcal{P}_c$ denotes the set of propagation paths whose integer delay and Doppler indices correspond to the c-th integer delay–Doppler tap, defined as

$$\mathcal{P}_c = \{i \in \{0, \ldots, P-1\} \mid (\ell_i, k_i) = (\ell^{(c)}, k^{(c)})\}.$$

The resulting channel representation is then fed into a three-layer multi-layer perceptron (MLP) (described in Table I) to generate the corresponding 1×1 convolution kernel.

2) Fractional Doppler Kernel Generation: For the fractional Doppler case, we first construct a channel gain-weighted per-tap fractional Doppler spread as

$$\tilde{h}_c[q] = \begin{cases} \sum_{i \in \mathcal{P}_c} \hat{h}_i \beta_i(q), & \mathcal{P}_c \neq \emptyset \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

where $q \in \{-N_i, \ldots, N_i\}$.

We convert this complex channel representation into a real-valued tensor $\mathbf{h}_{\mathrm{in}} \in \mathbb{R}^{2(2N_i+1) \times C}$, where the real and imaginary components are concatenated along the spatial dimension. The tensor then is processed by a two-layer MLP that learns an embedding of the complex Doppler spread function by mixing only the real and imaginary components independently across taps. The output is then reshaped into $\mathbb{R}^{1 \times (2N_i+1) \times 2C}$ and processed by a small CNN with 1×1 convolutions to capture interactions across feature channels. The architectural details are summarized in Table I. Here, C denotes the number of integer delay-Doppler channel taps defined in (15), while $C^{\mathcal{G}}_{\mathrm{in}}$ denotes the input channel dimension of the RDN $\mathcal{G}_\phi(\cdot)$ in (21), which processes the feature map resulting from the Align-Combine module.

With the channel-conditioned kernels generated for both the integer and fractional Doppler cases, we now describe their integration into the overall initialization pipeline. The extracted kernels are convolved with the extended input $\mathbf{Y}_{\mathrm{ext}}$ to obtain channel-conditioned features, which are subsequently processed by the RDN to produce the initial detection state $\mathbf{Z}^{(0)}_{\mathrm{state}}$. The classifier head then maps $\mathbf{Z}^{(0)}_{\mathrm{state}}$ to the initial symbol estimates $\hat{\mathbf{p}}^{(0)}$. The classifier head is described in Table II. Here, C_{state} represents the channel dimension of the detection state. In our experiments, we set $C_{\mathrm{state}} = C^{\mathcal{G}}_{\mathrm{in}} = 128$.

TABLE I
CHANNEL-AWARE KERNEL EXTRACTORS FOR INTEGER AND FRACTIONAL DOPPLER CASES

Integer Doppler (MLP)				**Fractional Doppler (MLP + CNN)**			
Layer	**Input**	**Output**	**Activation**	**Layer**	**Input**	**Output**	**Activation**
Linear	$2C$	128	LeakyReLU	Linear	$2(2N_i+1)$	64	LeakyReLU
Linear	128	128	LeakyReLU	Linear	64	$2(2N_i+1)$	LeakyReLU
Linear	128	$2C \cdot C_{\text{in}}^{\mathcal{G}}$	–	Reshape	$2(2N_i+1) \times C$	$1 \times (2N_i+1) \times 2C$	–
–	–	–	–	Conv 1×1	$2C$	128	LeakyReLU
–	–	–	–	Conv 1×1	128	128	LeakyReLU
–	–	–	–	Conv 1×1	128	$2C \cdot C_{\text{in}}^{\mathcal{G}}$	–

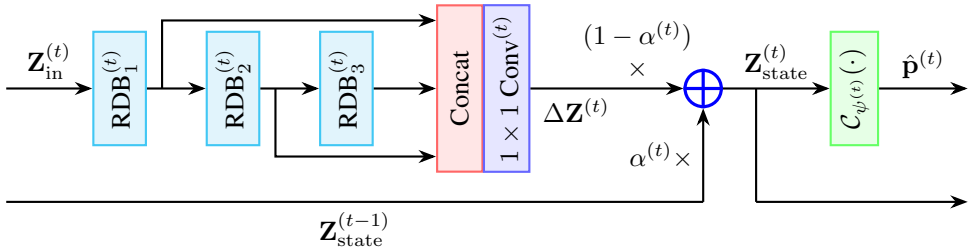

Fig. 3. Iteration t of the proposed IRN. The figure illustrates the refinement module, the detection state update rule, and the classification head for symbol detection.

TABLE II
CLASSIFIER HEAD

Layer	**Input**	**Output**	**Activation**
Conv 1×1	C_{state}	64	LeakyReLU
Conv 1×1	64	Q	–

TABLE III
RDN FOR INITIAL SYMBOL ESTIMATION

Layer	**# Conv. Layers**	**Input**	**Growth Rate**	**Kernel Size**
RDB	5	$C_{\text{in}}^{\mathcal{G}}$	64	3×3
RDB	5	64	64	3×3
RDB	5	64	64	3×3
RDB	5	64	64	3×3

The overall module can be summarized as

$$\mathbf{Y}_{\text{ext}} = \mathcal{T}(\mathbf{Y}) \tag{19}$$

$$\mathbf{W} = f_\theta(\mathbf{h}_{\text{in}}) \tag{20}$$

$$\mathbf{Z}_{\text{state}}^{(0)} = \mathcal{G}_\phi(\mathcal{F}_{\mathbf{W}}(\mathbf{Y}_{\text{ext}})) \tag{21}$$

$$\hat{\mathbf{p}}^{(0)} = \mathcal{C}_{\psi^{(0)}}(\mathbf{Z}_{\text{state}}^{(0)}), \tag{22}$$

where $\mathcal{T}(\cdot)$ represents the alignment stage of the Align-Combine, $f_\theta(\cdot)$ denotes the channel-conditioned kernel generation module, $\mathcal{F}_{\mathbf{W}}(\cdot)$ represents the convolution operator parameterized by $\mathbf{W}$, $\mathcal{G}_\phi(\cdot)$ denotes the RDN-based network (see Table III), and $\mathcal{C}_{\psi^{(0)}}(\cdot)$ is the classifier head.

B. Iterative Refinement Network

While the channel-conditioned initialization provides an informed starting point, it remains a single-shot estimate and may retain residual errors caused by noise and interference. This motivates an iterative refinement stage that progressively updates the detection result, allowing information from previous estimates and reconstructed signals to be accumulated and used for subsequent corrections. We therefore design the IRN to maintain and update a detection state across multiple refinement steps. At each iteration t, the detection state $\mathbf{Z}_{\text{state}}^{(t)}$ is updated through a learned residual correction (see Fig. 3) and used to generate refined symbol probabilities, which serve as input to the next iteration. The parameters of the refinement network are not shared across iterations, allowing each iteration to learn a distinct residual correction. Through this iterative process, the detector can successively correct errors remaining from earlier iterations.

The input at iteration t consists of several components. First, we include the predicted symbol probabilities from the previous iteration, $\hat{\mathbf{p}}^{(t-1)} \in \mathbb{R}^{M\times N\times Q}$ which carry forward the current symbol-level detection beliefs into the next refinement step. Then, based on these probabilities, soft symbol estimates are computed as

$$\hat{X}_{\text{DD}}^{(t)}[\ell,k] = \sum_{q=0}^{Q-1} a_q \hat{p}^{(t-1)}[\ell,k,q]. \tag{23}$$

where $a_q \in \mathbb{C}$ denotes the q-th symbol in $\mathcal{A}$.

Using these complex soft estimates, we reconstruct the received signal according to the system model introduced in Section II, representing its real and imaginary components as $\hat{\mathbf{Y}}^{(t)} \in \mathbb{R}^{M\times N\times 2}$, as well as an extended per-tap reconstruction $\hat{\mathbf{Y}}_{\text{ext}}^{(t)} \in \mathbb{R}^{M\times N\times 2C}$. This per-tap reconstruction induces a fixed delay–Doppler representation, where each feature channel corresponds to a predefined shift index associated with a specific channel tap. This enables a consistent association between feature channels and physical shifts, allowing the model to internalize each tap's contribution in a structured manner. The reconstructed signal is then used to compute the reconstruction error, $\mathbf{E} = \mathbf{Y} - \hat{\mathbf{Y}}$.

Beyond the per-tap reconstruction, the original received signal $\mathbf{Y}$ is retained explicitly, since $\hat{\mathbf{Y}}_{\text{ext}}^{(t)}$ is a function of the current symbol estimates and channel state and may not preserve all information present in the observation. Including $\mathbf{Y}$ directly guards against this potential information loss. The residual $\mathbf{E}$ is provided explicitly rather than left for the network to infer, giving the refinement network direct access to the observation–reconstruction mismatch without requiring it to relearn this subtraction from $\mathbf{Y}$ and $\hat{\mathbf{Y}}_{\text{ext}}^{(t)}$ at every iteration. Its magnitude $|\mathbf{E}|$ further highlights regions where the mismatch is most pronounced, providing an explicit measure of error severity.

The full input to the refinement layer is therefore given by

$$\mathbf{Z}_{\text{in}}^{(t)} = [\hat{\mathbf{p}}^{(t-1)},\, \hat{\mathbf{Y}}_{\text{ext}}^{(t)},\, \mathbf{Y},\, \mathbf{E},\, |\mathbf{E}|] \in \mathbb{R}^{M\times N\times(Q+2C+5)}. \tag{24}$$

Residual dense blocks (RDBs) [29] are well suited for the refinement stages as their dense and residual connections naturally accommodate the joint processing of heterogeneous inputs such as previous stage symbol probabilities, reconstructed signal features, and the received signal. These connections promote effective feature reuse, ensuring that information from these inputs remains accessible throughout the block. This helps preserve useful low-level details while enabling progressive refinement of the latent representation for improved OTFS detection. For this reason, we employ three RDBs in the refinement layer, concatenate their outputs, and fuse them using a 1×1 convolution to generate a refinement update. The updated detection state is obtained by scaling and combining the previous detection state and the refinement update, where a learnable scalar $\alpha^{(t)}$ adaptively balances their contributions at each iteration. The update is given by

$$\mathbf{Z}_{\text{state}}^{(t)} = \alpha^{(t)}\mathbf{Z}_{\text{state}}^{(t-1)} + (1-\alpha^{(t)})\Delta\mathbf{Z}^{(t)}. \tag{25}$$

This updated detection state is then fed into the classifier head $\mathcal{C}_{\psi^{(t)}}(\cdot)$ to produce the symbol probability estimates $\hat{\mathbf{p}}^{(t)}$. The classifier heads in the iterative refinement network have the same architecture and number of parameters as those in the initial estimation stage (see Table II).

C. Training

The proposed network is trained using synthetically generated OTFS frames under the doubly-dispersive channel model described in Section II. At each training epoch, 1800 channel realizations are generated, with 65 OTFS frames per channel realization, where the transmitted symbols for each frame are uniformly sampled from the modulation alphabet. The signal-to-noise ratio (SNR) for each frame is uniformly sampled from 5–25 dB. The network is trained using the Adam optimizer with a cosine learning-rate schedule.

The training objective includes supervision at both the initialization stage and all subsequent refinement stages. Specifically, a cross-entropy loss is applied to the symbol probability estimates produced at each iteration. The overall training objective is given by

$$L = \sum_{t=0}^{T} \mathcal{L}_{\text{CE}}\left(\hat{\mathbf{p}}^{(t)}, \mathbf{y}_{\text{gt}}\right), \tag{26}$$

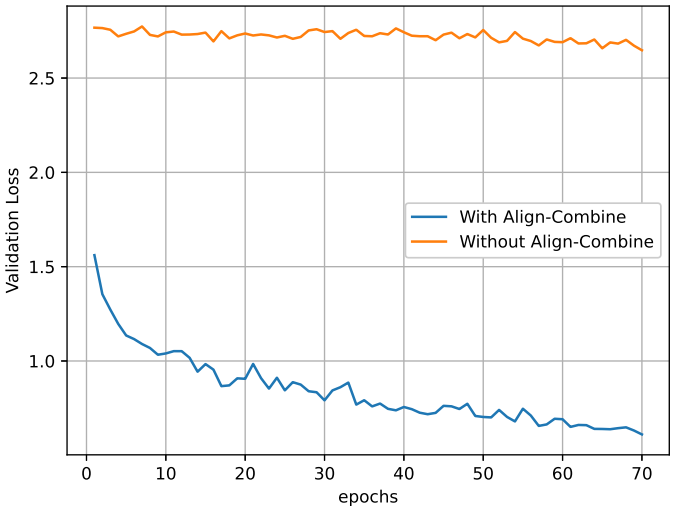

Fig. 4. Cross-entropy loss of the initialization stage with and without the Align–Combine module for 16-QAM.

where $\hat{\mathbf{p}}^{(t)}$ denotes the predicted symbol probability distribution at iteration t, $\mathbf{y}_{\text{gt}}$ represents the ground-truth symbol labels, $\mathcal{L}_{\text{CE}}(\cdot,\cdot)$ denotes the cross-entropy loss, and T is the total number of refinement iterations. We use uniform weighting across all stages in (26), encouraging the detector to produce accurate estimates throughout the iterative refinement process without explicitly prioritizing any particular stage.

IV. Simulation Results

To assess the performance of our proposed receiver, we simulate an OTFS system with carrier frequency $f_{\text{c}} = 4$ GHz, subcarrier spacing $\Delta f = 15$ kHz, $M = 32$ delay bins, $N = 32$ Doppler bins. Unless otherwise specified, the channel consists of $P = 5$ multipath components, with integer delay indices uniformly drawn from $[0, \ell_{\max}]$, where $\ell_{\max} = 3$; Doppler shifts are generated according to Jakes' model with maximum Doppler shift $k_{\max} = 2$; the channel gains are distributed as $\mathcal{CN}(0, 1/P)$.

A. Architectural Evaluation

We first demonstrate that removing the Align-Combine module severely limits the RDN's ability to generalize to unseen channels. To isolate this effect, we evaluate the initialization stage using the same RDN architecture but feed it the raw received signal $\mathbf{Y} \in \mathbb{R}^{M\times N\times 2}$ without any alignment or channel-conditioned combination. The training and validation sets are generated using disjoint channel realizations. As shown in Fig. 4, the RDN fails to generalize to unseen channels when the Align-Combine module is removed, confirming that channel-aware feature construction is essential for robust initialization.

To evaluate the effectiveness of the proposed channel-conditioned initialization stage, we compare the full two-stage model against an IRN-only variant (labeled as *IRN* in Fig. 5) initialized with uniform symbol probabilities. The proposed initializer provides an informative prior, enabling the refinement network to start from a more reliable detection

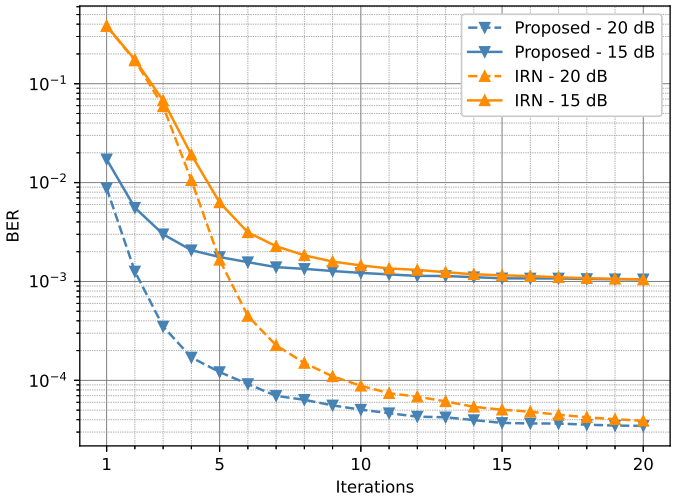

Fig. 5. BER versus refinement iterations for the proposed method and the IRN-only baseline under 16-QAM modulation with fractional Doppler at E_b/N_0 = 15 dB and E_b/N_0 = 20 dB.

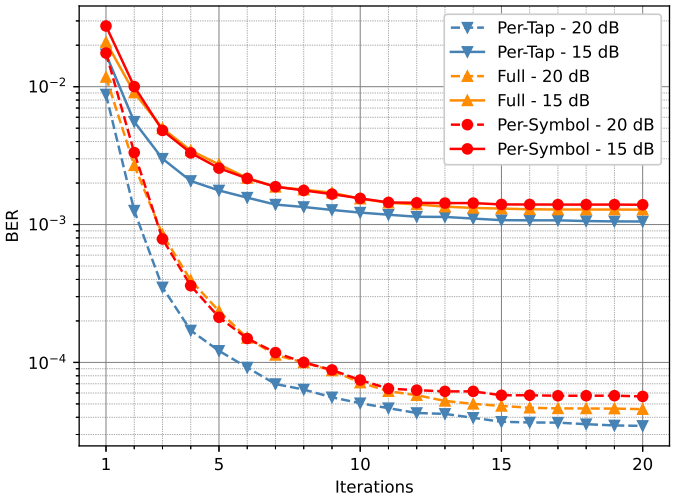

Fig. 6. BER versus refinement iterations for the proposed method under different signal reconstruction decompositions, evaluated for 16-QAM modulation with fractional Doppler at E_b/N_0 = 15 dB and E_b/N_0 = 20 dB.

state. As shown in Fig. 5, the proposed framework achieves substantially faster convergence at both SNR levels (15 dB and 20 dB), obtaining lower BER during the early refinement iterations. Although both variants gradually approach similar error floors at higher iteration counts, the proposed initialization reduces the number of refinement iterations required to achieve a given BER level.

For further investigation, we study the effect of the signal reconstruction decomposition on the convergence behavior and detection performance of the proposed framework. We experiment with three variants. The first is the full reconstruction ($\hat{\mathbf{Y}}^{(t)} \in \mathbb{R}^{M\times N\times 2}$), where the real and imaginary components of the reconstructed signal are represented as separate channels of a two-channel tensor. The second is the per-tap decomposition ($\hat{\mathbf{Y}}^{(t)}_{\text{ext}} \in \mathbb{R}^{M\times N\times 2C}$), which separates contributions across delay–Doppler channel taps. Finally, the per-symbol decomposition ($\hat{\mathbf{Y}}^{(t)}_{\text{sym}} \in \mathbb{R}^{M\times N\times 2P(2N_i+1)}$), which is inspired by the connectivity structure underlying MP detection [7], separates each symbol's contribution to the received delay–Doppler grid into its own feature map channel.

As shown in Fig. 6, the per-tap decomposition achieves both faster convergence and a lower BER floor compared to the full and per-symbol reconstructions at both SNR levels. This can be attributed to the structured nature of the per-tap representation, where each feature channel corresponds to a specific delay–Doppler tap, maintaining a fixed mapping between feature channels and the underlying delay–Doppler channel structure. These results indicate that providing the proposed IRN with a decomposition that preserves the delay–Doppler structure is important for effective iterative refinement.

B. Detection Performance

In this subsection, we compare the proposed iterative detector with conventional model-based detectors, namely MMSE and MP, as well as representative single-shot deep learning detectors. Specifically, we include an RDN with the same

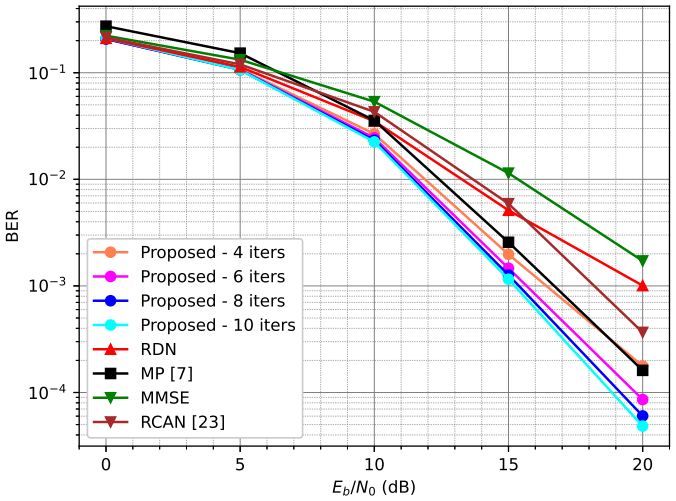

Fig. 7. BER performance comparison under integer Doppler for 16-QAM.

number of residual dense blocks as the four-iteration version of the proposed detector to represent a purely CNN-based single-shot approach, and the RCAN-based detector [23] as a representative hybrid single-shot detector[1] that utilizes an MMSE estimate as auxiliary information. The performance of all detectors is evaluated under channels with both integer and fractional Doppler shifts.

In Fig. 7, we compare the BER performance of the proposed detector and the baseline methods under integer Doppler conditions. The proposed detector exhibits consistent improvements as the number of iterations increases. With four iterations, it outperforms MP in the low-to-moderate SNR regime and approaches its performance at high SNR, where MP becomes slightly stronger and outperforms MMSE, RDN, and RCAN. Among the single-shot detectors, the hybrid RCAN consistently improves upon its MMSE initialization across

[1]We have also tested the 2D-CNN architecture in [19] to represent the hybrid single-shot CNN approach. However, it yields nearly identical performance to MP, and is therefore omitted from the reported results.

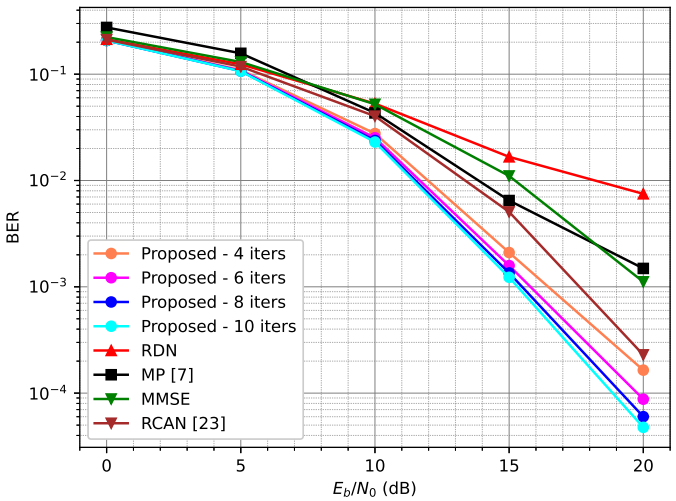

Fig. 8. BER performance comparison under fractional Doppler for 16-QAM.

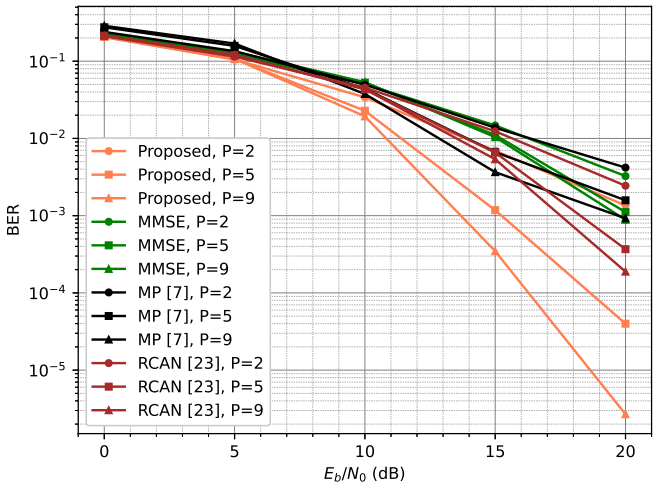

Fig. 9. BER performance comparison under varying numbers of channel paths. The proposed model results are shown for 10 iterative refinement iterations.

the entire SNR range. In contrast, RDN initially outperforms RCAN at low SNR, but RCAN eventually surpasses RDN as SNR increases. Increasing the number of iterations beyond four enables the proposed detector to outperform all baseline methods across the entire SNR range.

In Fig. 8, we evaluate the proposed detector under fractional Doppler conditions. While the single-shot RDN outperforms conventional methods such as MMSE and MP at low SNR, its performance degrades significantly in the mid-to-high SNR regime. This suggests that although a single-shot CNN can effectively suppress noise, it is less effective at mitigating the interference caused by fractional-Doppler leakage as noise becomes less dominant. The RCAN-based detector consistently improves upon the MMSE prior across all SNR values, confirming the effectiveness of learned refinement over conventional estimation. However, it remains a single-shot approach that produces its output in a single pass conditioned on the MMSE estimate, without the ability to iteratively revisit or refine that output. In contrast, the proposed detector with four iterations achieves consistent performance gain over the single-shot RDN, RCAN and classical MP and MMSE detectors. In addition, increasing the number of iterations of the proposed method consistently improves performance, achieving progressively lower BER across the entire SNR range. This trend highlights the benefits of iterative refinement in mitigating doubly-dispersive channel induced interference.

Comparing the two Doppler regimes, we observe that classical MMSE detection exhibits a slight performance improvement under fractional Doppler. Since RCAN operates on an augmented input comprising the received signal and the initial MMSE estimate, its performance improves accordingly. In contrast, MP detection performs very well under integer Doppler, but its performance degrades under fractional Doppler. The proposed iterative detector maintains nearly identical performance across both regimes, indicating robustness to the energy spread introduced by fractional Doppler.

C. Robustness to Multipath Variations

We further evaluate the proposed detector under varying numbers of multipath components (Fig. 9). For this experiment, the proposed model was trained using channel realizations whose number of multipath components were uniformly sampled between 2 and 10, and evaluted at $P \in \{2, 5, 9\}$. At high SNR, all detectors exhibit worse performance with only $P = 2$ separable paths than $P = 5$ and $P = 9$ due to reduced multipath diversity [30], [31]. As the number of multipath components increases, all detectors benefit from the additional diversity to varying degrees. MMSE shows only marginal improvement, consistent with its limited capacity to resolve the increasingly dense interference structure introduced by additional paths. MP and RCAN show moderate gains as P increases from 5 to 9, particularly at high SNR. The proposed detector, however, exhibits a substantially larger relative improvement over the same range, most pronounced at high SNR where the residual error is dominated by unresolved multipath interference rather than noise.

A comparison with RCAN provides further insight into the role of iterative refinement. While RCAN uses the MMSE symbol estimate concatenated with the raw received signal as auxiliary information and produces a symbol estimate in a single inference pass, the proposed detector performs iterative refinement over multiple stages. As the number of propagation paths increases, the proposed detector exhibits substantially larger performance gains than RCAN, suggesting that iterative refinement provides greater benefit in exploiting the additional multipath diversity than a single-shot detector.

Fig. 9 also shows that the proposed neural detector exhibits robustness to variation in the number of multipath components. Despite being trained under a mixed regime of channel path counts, the proposed model achieves BER performance comparable to the configuration trained using a fixed $P = 5$ (Fig. 8), which serves as the main reference training setup used in earlier experiments. In addition, it consistently outperforms the considered baselines across the reported range of P,

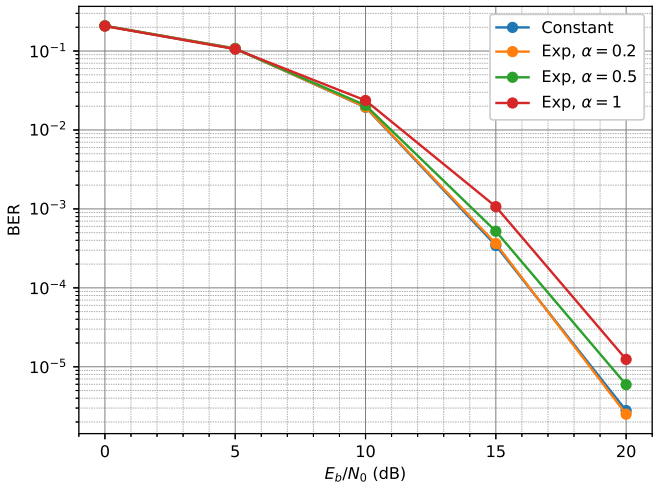

Fig. 10. BER performance comparison for constant and exponential power delay profiles with varying decay factors α. Results are obtained using 10 iterative refinement iterations.

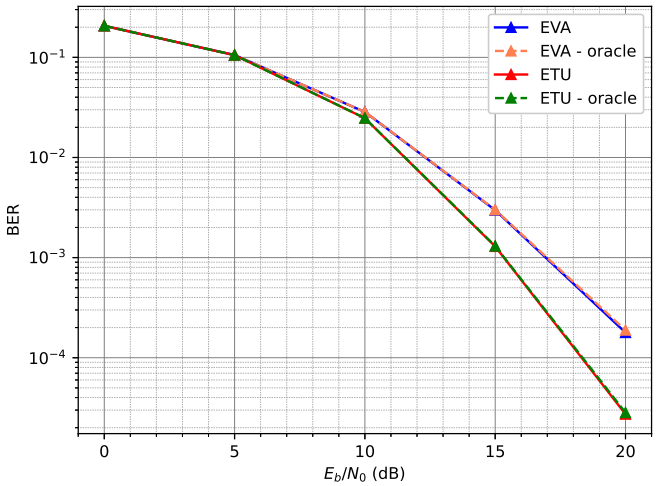

Fig. 11. BER performance on EVA and ETU channels for the proposed detector under zero-shot generalization, compared against oracle baselines trained directly on the respective channel.

demonstrating suitability for practical scenarios where the number of multipath components is unknown a priori and may vary over time.

D. Robustness to Power Delay Profile Variations

To evaluate the robustness of the proposed model across diverse channel conditions, the model is trained using synthetic channels with Rayleigh fading generated from two power delay profiles (PDPs): a constant PDP and a normalized exponential PDP characterized by a decay factor α [32]. For each channel realization, α is randomly sampled from the range $[0.1, 1]$ to provide diversity in the delay power distribution.

Fig. 10 illustrates the BER performance of the proposed model under a constant PDP and exponential PDPs with fixed decay factors of $\alpha \in \{0.2, 0.5, 1\}$. These decay factors are selected from the range used during training. It can be observed that the proposed model reliably detects the transmitted information bits across all tested channel conditions. The gradual degradation in performance with increasing α can be attributed to the reduction in effective multipath diversity, as larger decay factors concentrate more channel energy in the early delay taps and diminish the contribution of weaker multipath components.

E. Generalization to Standardized Channel Models

We also investigate the performance of the proposed model on standardized Extended Vehicular A (EVA) and Extended Typical Urban (ETU) channel models published by the 3rd Generation Partnership Project (3GPP) in [33]. To assess its ability to generalize to unseen channel distributions, we compare the proposed model trained on the synthetic channel dataset with models trained exclusively on EVA and ETU channels, denoted as *EVA-oracle* and *ETU-oracle*, respectively. As shown in Fig. 11, the proposed model achieves comparable BER performance to the oracle models across the evaluated SNR range and slightly outperforms them at high SNRs, further demonstrating our model's ability to generalize to previously unseen channel distributions.

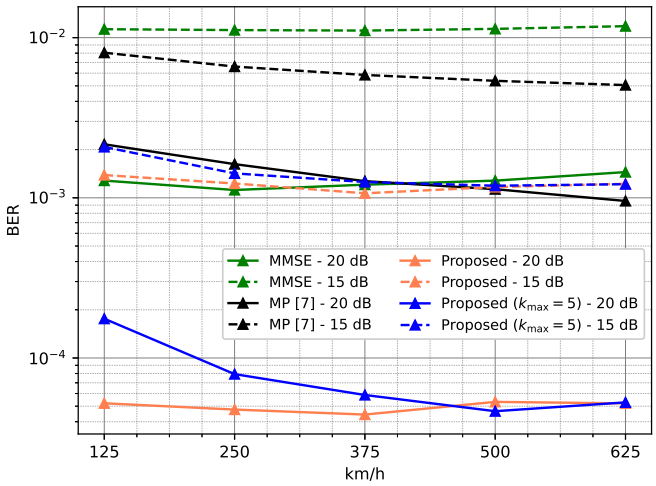

Fig. 12. BER performance comparison as a function of user velocity at $E_b/N_0 = 15$ dB and 20 dB. The selected velocities correspond to maximum normalized Doppler shift indices of $k_{\max} = 1, 2, 3, 4, 5$. The proposed model results are shown for 10 iterative refinement iterations.

F. Robustness to Doppler Variations

In Fig. 12, we evaluate the performance of the considered detectors under varying user velocities (equivalently, maximum Doppler shifts). Consistent with the assumption in Section III that the maximum delay and Doppler spreads are known, the curve labeled *Proposed* corresponds to an instance of the proposed detector trained for the corresponding maximum Doppler shift. As shown, the proposed detector maintains nearly constant BER across the considered velocity range at both $E_b/N_0 = 15$ dB and 20 dB. The MMSE detector also exhibits relatively stable performance with changing velocity,

TABLE IV
RUNTIME AND BER COMPARISON OF MMSE, MP, AND THE PROPOSED METHOD ACROSS DIFFERENT ITERATION BUDGETS AT 20 DB SNR

Metric	**MMSE**	**MP**			**Proposed**		
		5 iters	**10 iters**	**15 iters**	**5 iters**	**10 iters**	**15 iters**
Inference Time (ms/batch, $B = 1$**)**	7.388	93.634	111.457	120.898	61.370	108.800	149.557
Inference Time (ms/batch, $B = 32$**)**	108.633	292.092	418.352	549.431	61.779	106.587	150.075
BER	1.1×10^{-3}	2.2×10^{-1}	1.8×10^{-2}	2.3×10^{-3}	1.2×10^{-4}	4.8×10^{-5}	3.4×10^{-5}

whereas the MP detector benefits from increasing Doppler spread, yielding improved BER as velocity increases.

We further evaluate the generalization capability of the proposed detector by considering a single model trained only for channels with a maximum normalized Doppler shift of $k_{\max} = 5$, labeled *Proposed (*$k_{\max} = 5$*)*. When evaluated on channels with lower maximum Doppler shifts, this model achieves performance comparable to the velocity-specific model for neighboring Doppler regimes, while a more noticeable performance degradation is observed for $k_{\max} = 2$ and $k_{\max} = 1$. These results indicate that a model trained for a single maximum Doppler condition can generalize well to nearby Doppler regimes, which is attractive in practical scenarios where the maximum Doppler shift is not known a priori and may vary over time.

G. Complexity

In this section, we evaluate the performance and computational efficiency of the considered detection algorithms through numerical simulations. The analytical FLOPs derivation is provided in the Appendix. All runtime measurements are obtained using a batch size of one over 10^4 independently generated OTFS test frames to ensure statistically reliable estimates. The simulations are carried out on a Google Colab environment equipped with an NVIDIA A100 GPU. For the MP detector, we employ a vectorized implementation in which MP updates are fully vectorized and iteration-independent computations are precomputed and reused across iterations. This ensures a consistent and efficient evaluation of the MP baseline, enabling a fair runtime comparison with the proposed method. The MMSE detector is also implemented using PyTorch tensor operations to ensure a consistent GPU-based evaluation framework across all methods.

From Table IV, we observe that under single-sample inference ($B = 1$), the MP runtime exhibits sub-linear growth with respect to the number of iterations. This behavior is influenced by fixed implementation overheads and the early-stopping criterion, which terminates the iterative process once a convergence threshold is met. In contrast, the proposed method exhibits an approximately linear increase in runtime, reflecting a uniform per-iteration computational cost. As a result, MP has a higher runtime than the proposed method at 5 iterations, while the proposed method exceeds MP's runtime beyond 10 iterations. Crucially, the proposed method achieves substantially lower BER across all iteration counts and exhibits faster convergence behavior. At 5 iterations, it already reaches a BER of 1.2×10^{-4}, which is close to its saturation level, with only marginal improvement at higher iteration counts. In contrast, the MP detector improves more gradually, decreasing from 2.2×10^{-1} at 5 iterations to 2.3×10^{-3} at 15 iterations, indicating that it requires significantly more iterations to approach convergence. This demonstrates a more favorable complexity–performance tradeoff for the proposed method.

The batched inference results ($B = 32$) further highlight the computational characteristics of the proposed method. While the runtimes of both the MP and MMSE detectors increase substantially with batch size, the proposed method maintains nearly identical runtime for $B = 1$ and $B = 32$, demonstrating effective utilization of GPU parallelism. For instance, at 10 iterations, the proposed method requires 106.6 ms per batch for $B = 32$ compared to 108.8 ms for $B = 1$, whereas the MP detector increases from 111.5 ms to 418.4 ms. Similar trends appear across all iteration budgets. These results indicate that the proposed architecture scales efficiently with batch size and can process multiple OTFS frames simultaneously with minimal additional runtime overhead.

V. CONCLUSION

In this paper, we proposed a two-stage learnable receiver for robust OTFS detection. The key innovation in the proposed Align-Combine module is the explicit incorporation of the known OTFS delay–Doppler input–output relationship, combined with channel-conditioned convolutional kernels into the feature extraction process, improving robustness and the ability to generalize to unseen channel realizations. Furthermore, the proposed IRN demonstrates the effectiveness of iterative CNN-based state refinement for suppressing multipath interference, yielding improved BER performance, particularly in the mid-to-high SNR regime. Extensive simulations demonstrate that the proposed receiver remains effective across a wide range of channel conditions, including varying user velocities and multipath environments, while maintaining strong performance on previously unseen channel distributions.

Future work could extend the proposed iterative receiver to jointly refine channel and symbol estimates. The current framework treats the available channel estimate as fixed during iterative detection. A joint refinement architecture could instead maintain separate channel and detection states, with the channel state updated using pilot observations and feedback from the evolving symbol estimates, while the refined channel estimate is used to improve subsequent symbol detection. This would enable iterative information exchange between channel estimation and symbol detection within a unified refinement framework.

TABLE V
COMPUTATIONAL COMPLEXITY (FLOPS) COMPARISON OF DETECTION ALGORITHMS

Method	**Component**	**FLOPs** ($\times 10^9$)
Proposed	Align-Combine	0.2488
	RDN + Classifier	5.18
	IRN (T iterations)	$T \cdot 3.6$
	Total Proposed	$5.43 + T \cdot 3.6$
MP	MP Update (T iterations)	$T \cdot 0.034$
MMSE	MMSE Equalization (one-time)	11.47

APPENDIX

In this section, we analyze the computational complexity of the proposed method and compare it with conventional baseline algorithms, including MP and MMSE detectors. The analysis is performed in terms of floating-point operations (FLOPs), where each FLOP corresponds to a single real-valued multiplication or addition. Complex-valued operations are decomposed into real arithmetic, with each complex multiplication counted as 4 real multiplications and 2 real additions, and each complex addition counted as 2 real additions.

The Align-Combine module consists of two steps. The alignment step applies circular shifts to the received signal for all C integer delay–Doppler taps, where each shift requires computing MN indices at a cost of 2 FLOPs per index, yielding $2MNC$ FLOPs in total. The indexing itself is $\mathcal{O}(1)$.

The combine step involves three components. First, constructing the fractional Doppler input representation via the channel gain-weighted spread in (18) costs $10P(2N_i + 1)$ FLOPs across all P paths and $2N_i+1$ fractional bins. Second, the kernel generation network for the fractional Doppler case consists of an MLP and a CNN, costing $512(2N_i + 1)C$ and $256(2N_i+1)(2C+128+2CC_{\text{in}}^{\mathcal{G}})$ FLOPs, respectively. Finally, convolving the generated kernels with the extended input $\mathbf{Y}_{\text{ext}}$ over the $M \times N$ delay-Doppler grid costs $4CC_{\text{in}}^{\mathcal{G}}MN(2N_i+1)$ FLOPs.

Before analyzing the RDN, we first derive the FLOPs for a single RDB, which serves as the basic building block. The RDB consists of densely connected convolutional layers followed by a local feature fusion step, with a residual connection aggregating the input and output. The total FLOPs for a single RDB over an $M \times N$ feature map are denoted as $\text{FLOPs}_{\text{RDB}}$ and given by

$$\begin{aligned}\text{FLOPs}_{\text{RDB}} &= 2k_s^2MNG\sum_{i=0}^{L-1}(C_{\text{in}}^{\text{RDB}} + iG) \\ &\quad + 2C_{\text{out}}^{\text{RDB}}(C_{\text{in}}^{\text{RDB}} + LG)MN \\ &= 2k_s^2MNG(LC_{\text{in}}^{\text{RDB}} + G\frac{L(L-1)}{2}) \\ &\quad + 2C_{\text{out}}^{\text{RDB}}(C_{\text{in}}^{\text{RDB}} + LG)MN\end{aligned} \tag{27}$$

where $C_{\text{in}}^{\text{RDB}}$, G, $C_{\text{out}}^{\text{RDB}}$ denote the input channel size, growth rate and the output channel size of the RDB, respectively. Using this, the RDN consists of N_{RDB} stacked RDB blocks followed by a feature fusion convolution, costing $\sum_{i=1}^{N_{\text{RDB}}} \text{FLOPs}_{\text{RDB}_i} + 2MN(N_{\text{RDB}}C_{\text{out}}^{\text{RDB}})C_{\text{state}}$ FLOPs, where C_{state} represents the channel dimension of the detection state used in the iterative refinement process.

Finally, the classifier head costs $128MN(C_{\text{state}} + Q)$ FLOPs.

We next analyze the second stage of the proposed method, namely the IRN. The computational cost of a single iteration consists of five main components. First, soft symbol estimation requires $4MNQ$ FLOPs. Second, signal reconstruction incurs a cost of $10MN(2N_i + 1)P$ FLOPs. Third, feature refinement is performed through three RDBs followed by feature fusion convolution, with total complexity given by $\text{FLOPs}_{\text{RDB}_1} + \text{FLOPs}_{\text{RDB}_2} + \text{FLOPs}_{\text{RDB}_3} + 2MN(3C_{\text{out}}^{\text{RDB}})C_{\text{state}}$. Fourth, the gated state update introduces an additional cost of $3MNC_{\text{state}}$ FLOPs. Finally, the classifier head requires $128MN(C_{\text{state}} + Q)$ FLOPs.

For comparison, we now consider the computational complexity of the MP detector. The analysis is based on the optimized implementation used throughout this work, in which all quantities independent of the iterative message-passing updates are precomputed once and reused across iterations. This optimization does not alter the underlying MP algorithm or its asymptotic complexity. The FLOP count reported below therefore corresponds solely to the per-iteration message-passing stage, excluding one-time initialization overhead.

Let $\tilde{P}$ denote the number of non-zero entries in each row (or equivalently column) of the OTFS channel matrix, where $\tilde{P} \leq P(2N_i+1)$. In each MP iteration, the mean and variance updates require $2MN\tilde{P}(5+2Q)$ and $MN\tilde{P}(2Q+8)$ FLOPs, respectively. The probability update step incurs an additional $15MN\tilde{P}Q$ FLOPs. Therefore, the total computational complexity of a single MP iteration is obtained by summing these three terms.

The complexity of MMSE equalization is dominated by the complex matrix multiplication and inversion. The total FLOPs can be approximated as $\frac{32}{3}(MN)^3 + 16(MN)^2 + 2MN$.

For a quantitative comparison, the numerical FLOP counts obtained from the above complexity expressions are summarized in Table V. The reported values are computed using the parameter configuration adopted throughout this work, namely $M = N = 32$, $Q = 16$ (16-QAM), $N_i = 10$, and $P = 5$, together with the network hyperparameters specified in Section III.

REFERENCES

[1] Z. Wei, W. Yuan, S. Li, J. Yuan, G. Bharatula, R. Hadani, and L. Hanzo, "Orthogonal time-frequency space modulation: A promising next-generation waveform," *IEEE Wireless Communications*, vol. 28, no. 4, pp. 136–144, 2021.

[2] J. Wu and P. Fan, "A survey on high mobility wireless communications: Challenges, opportunities and solutions," *IEEE Access*, vol. 4, pp. 450–476, 2016.

[3] T. Wang, J. Proakis, E. Masry, and J. Zeidler, "Performance degradation of ofdm systems due to doppler spreading," *IEEE Transactions on Wireless Communications*, vol. 5, no. 6, pp. 1422–1432, 2006.

[4] J. G. Proakis and M. Salehi, *Digital Communications*, 5th ed. McGraw-Hill, 2008.

[5] R. Hadani, S. Rakib, M. Tsatsanis, A. Monk, A. J. Goldsmith, A. F. Molisch, and R. Calderbank, "Orthogonal time frequency space modulation," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, 2017, pp. 1–6.

[6] P. Raviteja, K. T. Phan, and Y. Hong, "Embedded pilot-aided channel estimation for otfs in delay–doppler channels," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 4906–4917, 2019.

[7] P. Raviteja, K. T. Phan, Y. Hong, and E. Viterbo, "Interference cancellation and iterative detection for orthogonal time frequency space modulation," *IEEE Transactions on Wireless Communications*, vol. 17, no. 10, pp. 6501–6515, 2018.

[8] G. D. Surabhi, R. M. Augustine, and A. Chockalingam, "On the diversity of uncoded otfs modulation in doubly-dispersive channels," *IEEE Transactions on Wireless Communications*, vol. 18, no. 6, pp. 3049–3063, 2019.

[9] A. Chockalingam, S. K. Mohammed, and R. Hadani, *OTFS Modulation: Theory and Applications*. John Wiley & Sons, 2024.

[10] J. S. Yedidia, W. T. Freeman, and Y. Weiss, *Understanding belief propagation and its generalizations*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003, p. 239–269.

[11] K. P. Murphy, Y. Weiss, and M. I. Jordan, "Loopy belief propagation for approximate inference: an empirical study," in *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, ser. UAI'99. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, p. 467–475.

[12] W. Yuan, Z. Wei, J. Yuan, and D. W. K. Ng, "A simple variational bayes detector for orthogonal time frequency space (otfs) modulation," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 7, pp. 7976–7980, 2020.

[13] G. D. Surabhi and A. Chockalingam, "Low-complexity linear equalization for otfs modulation," *IEEE Communications Letters*, vol. 24, no. 2, pp. 330–334, 2020.

[14] T. Zou, W. Xu, H. Gao, Z. Bie, Z. Feng, and Z. Ding, "Low-complexity linear equalization for otfs systems with rectangular waveforms," in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2021, pp. 1–6.

[15] S. Tiwari, S. S. Das, and V. Rangamgari, "Low complexity lmmse receiver for otfs," *IEEE Communications Letters*, vol. 23, no. 12, pp. 2205–2209, 2019.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, p. 84–90, May 2017. [Online]. Available: https://doi.org/10.1145/3065386

[17] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[19] Y. K. Enku, B. Bai, F. Wan, C. U. Guyo, I. N. Tiba, C. Zhang, and S. Li, "Two-dimensional convolutional neural network-based signal detection for otfs systems," *IEEE Wireless Communications Letters*, vol. 10, no. 11, pp. 2514–2518, 2021.

[20] A. Singh, S. Sharma, K. Deka, and V. Bhatia, "Dl-based otfs signal detection in presence of hardware impairments," *IEEE Wireless Communications Letters*, vol. 12, no. 9, pp. 1533–1537, 2023.

[21] Y. Wang, Z. Zhang, H. Li, T. Zhou, and Z. Cheng, "Signal detection based on separable cnn for otfs communication systems," *Entropy*, vol. 27, no. 8, 2025. [Online]. Available: https://www.mdpi.com/1099-4300/27/8/839

[22] Y. Wu, M. Zhou, Y. Lin, and Z. Liao, "Signal detection method for otfs system based on feature fusion and cnn," *Electronics*, vol. 14, no. 20, 2025. [Online]. Available: https://www.mdpi.com/2079-9292/14/20/4041

[23] A. Singh, S. Sharma, K. Deka, M. Sharma, and D. B. da Costa, "Rcan based otfs signal detection in the presence of hardware impairments," in *2025 IEEE 36th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2025, pp. 1–6.

[24] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part VII*. Berlin, Heidelberg: Springer-Verlag, 2018, p. 294–310. [Online]. Available: https://doi.org/10.1007/978-3-030-01234-2_18

[25] Y. Gao, X. Xu, Y. Jin, W. Yuan, J. Zhang, and S. Xu, "Joint channel estimation and data detection for otfs systems: A lightweight deep learning framework with a novel data augmentation method," *IEEE Internet of Things Journal*, vol. 12, no. 18, pp. 38 464–38 481, 2025.

[26] Y. Zhang, Y. Wang, Y. Liu, L. Shi, and Y. Zang, "A deep learning receiver for underwater acoustic otfs communications with doppler squint effect," *IEEE Wireless Communications Letters*, vol. 14, no. 4, pp. 1179–1183, 2025.

[27] Y. Gong, Q. Li, F. Meng, X. Li, and Z. Xu, "Data-driven deep learning for otfs detection," *China Communications*, vol. 20, no. 1, pp. 88–101, 2023.

[28] Y. Hong, T. Thaj, and E. Viterbo, *Delay-Doppler Communications: Principles and Applications*. Academic Press, 2022.

[29] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 7, pp. 2480–2495, 2021.

[30] R. Hadani and A. Monk, "Otfs: A new generation of modulation addressing the challenges of 5g," 2018. [Online]. Available: https://arxiv.org/abs/1802.02623

[31] S. Li, J. Yuan, W. Yuan, Z. Wei, B. Bai, and D. W. K. Ng, "Performance analysis of coded otfs systems over high-mobility channels," *IEEE Transactions on Wireless Communications*, vol. 20, no. 9, pp. 6033–6048, 2021.

[32] A. F. Molisch, *Wireless Communications*, 2nd ed. John Wiley & Sons, 2011.

[33] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) radio transmission and reception," 3rd Generation Partnership Project, Technical Specification 36.101, Jul. 2025, version 18.10.0.