Receiver Processing Techniques for Zak-OTFS with Superimposed Spread Pilot: A Comparison

Fathima Jesbin and Ananthanarayanan Chockalingam
Department of ECE, Indian Institute of Science, Bangalore

***Abstract*—The performance and complexity of Zak-OTFS receiver relies heavily on the pilot frame structure and the pulse shaping filter. While guard-band-protected pilot schemes (e.g., exclusive or embedded pilots) simplify the receiver, they incur significant penalties in spectral efficiency (SE) and peak-to-average power ratio (PAPR). Superimposed spread pilot (SSP) offers a good alternative by superimposing data and spread pilot across the full delay-Doppler (DD) frame, thereby maximizing SE and lowering PAPR. However, this efficiency comes at the cost of pilot-data interference, rendering simple linear estimation ineffective. In this paper, we address this issue through a comprehensive comparative study of receiver processing techniques for Zak-OTFS with SSP under different pulse shaping filters, namely, sinc, Gaussian, and Gaussian-sinc (GS) filters. We evaluate four receiver architectures for Zak-OTFS with SSP: $i)$ a baseline cross-ambiguity readoff estimator, $ii)$ a turbo receiver, $iii)$ a CNN denoiser, and $iv)$ a proposed hybrid CNN-aided turbo receiver. Simulation results demonstrate that the receiver choice is influenced by the pulse shape. While the GS pulse allows effective estimation and detection using standard turbo processing, the Gaussian pulse requires learning-based denoising and the sinc pulse requires the proposed hybrid architecture to achieve performance close to its perfect CSI benchmark.**

***Index Terms*—Zak-OTFS modulation, delay-Doppler domain, pulse shape, superimposed spread pilot, turbo processing, CNN.**

I. Introduction

Zak transform based orthogonal time frequency space (Zak-OTFS) modulation, a modulation in the delay-Doppler (DD) domain, has attracted considerable interest due to its robustness in high-Doppler channels and its structural advantages over multicarrier OTFS (MC-OTFS) [1],[2],[3]. Zak-OTFS carries out DD domain to time domain conversion in a single step using inverse Zak transform at the transmitter (and vice versa using Zak transform at the receiver) [1]-[5]. In Zak-OTFS, the end-to-end DD input-output (I/O) relation (i.e., end-to-end effective channel which is a cascade of transmit filter, physical channel, and receive filter) admits a particularly simple form, expressed as a cascade of twisted convolutions, which is attractive for receiver signal processing. The structure renders the Zak-OTFS I/O relation predictable under crystalline condition (where the delay and Doppler periods of the waveform are greater than the maximum delay and Doppler spreads of the effective channel, respectively) and enables efficient equalization/detection [2].

A consequence of predictability in Zak-OTFS is that the I/O relation can be estimated in a model-free manner, without explicitly estimating the channel parameters (delays/Dopplers/gains), through a simple read-off operation of the received frame containing the pilot [2],[3]. However, the performance of this model-free approach is majorly influenced by the following two factors: $i)$ *the structure of the pilot frame used* (e.g., exclusive pilot [2],[3], embedded pilot [9],and superimposed spread pilot [10]-[12]), and $ii)$ the DD localization and orthogonality attributes of *the DD pulse shaping filter* (e.g., sinc [2]-[6],[8], root raised cosine (RRC) [2]-[6],[8], Gaussian [7],[8], and Gaussian-sinc (GS) [8] filters). The effect of these two factors and the motivation and contributions in this paper are highlighted below.

In exclusive and embedded pilot frames, the pilot and data symbols are isolated by guard space, which limits pilot-data interference. However, they incur significant penalties in spectral efficiency (SE) and peak-to-average power ratio (PAPR). Superimposed spread pilot (SSP) frame is a good alternative which superimposes a spread pilot over data symbols across the entire DD frame without any guard space, thereby maximizing SE and lowering PAPR [10]. However, this comes at the cost of pilot-data interference, which requires more sophisticated receiver processing beyond the simple model-free approach to deal with the pilot-data interference. Also, while the sinc pulse shaping filter is best in terms of orthogonality (with nulls at the DD sampling points) which allows good detection performance, its DD localization is poor (with high side-lobes) which significantly degrades I/O relation estimation performance, rendering the simple model-free read-off based estimation inadequate. These motivate the need for robust receiver processing techniques suited for Zak-OTFS with SSP and different pulse shaping filters.

Based on the above, in this paper, we consider and compare receiver techniques that can offer robust performance in Zak-OTFS with SSP for different pulse shaping filters. Our contributions in this paper can be summarized as follows.

- We consider four receiver architectures for Zak-OTFS with SSP, ranging from low-complexity linear methods to advanced learning-based and turbo methods. These architectures include: $i)$ a baseline read-off estimator based on cross-ambiguity function computation [10], $ii)$ a turbo receiver performing iterative estimation and interference cancellation [11], $iii)$ a convolutional neural network (CNN) receiver that uses learning-based denoising [12], and $iv)$ a proposed hybrid CNN-aided turbo receiver.
- Different pulse shaping filters require different levels of receiver sophistication to achieve performance close to

This work was supported in part by the J. C. Bose National Fellowship and the NSF-DST project (International Cooperation Division), Department of Science and Technology, Government of India.

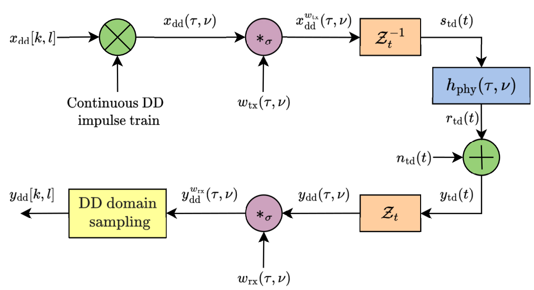


Fig. 1: Block diagram of Zak-OTFS transceiver.

their respective perfect channel state information (CSI) benchmarks. We show this through simulations for sinc, Gaussian, and GS filters in Veh-A channels [13] with fractional DDs.

- Our simulation results offer the following insights: 1) for all the considered filters, simple cross-ambiguity read-off based estimator is inadequate because of the untreated pilot-data interference, and subsequent processing is needed, 2) the GS filter, benefiting from its better orthogonality and localization, is robust enough to reach close to its perfect CSI benchmark with the turbo processing add-on (interference cancellation) to the cross-ambiguity read-off, 3) the Gaussian pulse, despite its excellent localization, requires learning-based denoising (CNN) to mitigate the effects of non-orthogonality, and 4) the sinc pulse, suffering from severe interference due to sidelobes despite its orthogonality, requires the proposed hybrid architecture (combination of cross-ambiguity, CNN denoising, and turbo) to reach close to its perfect CSI benchmark.

II. Zak-OTFS system model

Figure 1 shows the Zak-OTFS transceiver architecture. The Zak-OTFS carrier waveform is a DD domain quasi-periodic pulse defined by delay period τ_{p} and Doppler period $\nu_{\mathrm{p}} = 1/\tau_{\mathrm{p}}$. The fundamental DD period $\mathcal{D}_0 = \{(\tau,\nu) : 0 \le \tau < \tau_{\mathrm{p}}, 0 \le \nu < \nu_{\mathrm{p}}\}$ is discretized into the grid $\{(k\frac{\tau_{\mathrm{p}}}{M}, l\frac{\nu_{\mathrm{p}}}{N})|k \in \mathbb{K}, l \in \mathbb{L}\}$, where the index sets are given by $\mathbb{K} = \{0,\ldots,M-1\}$ and $\mathbb{L} = \{0,\ldots,N-1\}$. This corresponds to a frame limited to time duration $T = N\tau_{\mathrm{p}}$ and bandwidth $B = M\nu_{\mathrm{p}}$.

Transmitted signal: The transmitted signal comprises a superposition of data and spread pilot signals. Define the data symbols as $x[k,l] \in \mathbb{A}$ for $k \in \mathbb{K}, l \in \mathbb{L}$, normalized to unit average energy. These symbols are carried by the data signal

$$x_{\mathrm{d,dd}}[k,l] = \frac{1}{\sqrt{MN}} \sum_{k'=0}^{M-1} \sum_{l'=0}^{N-1} x[k',l'] \sum_{n,m\in\mathbb{Z}} e^{j2\pi\frac{nl'}{N}} \delta[k-k'-nM]\delta[l-l'-mN], \tag{1}$$

where $\delta[\cdot]$ is the discrete Dirac-delta function and $1/\sqrt{MN}$ ensures unit energy normalization.

The spread pilot signal is generated by spreading a unit energy point pilot $x_{\mathrm{p,dd}}[k,l] = \sum_{n,m\in\mathbb{Z}} e^{j2\pi\frac{nl_{\mathrm{p}}}{N}} \delta[k-k_{\mathrm{p}} - nM]\delta[l-l_{\mathrm{p}}-mN]$ located at $(k_{\mathrm{p}}, l_{\mathrm{p}})$ via an MN-periodic discrete twisted convolution[1] (denoted by $\circledast_{\sigma\mathrm{d}}$) with a discrete spreading filter $w[k,l]$:

$$\begin{aligned} x_{\mathrm{s,dd}}[k,l] &= w[k,l] \circledast_{\sigma\mathrm{d}} x_{\mathrm{p,dd}}[k,l] \\ &= \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} w[k-k_{\mathrm{p}}-nM, l-l_{\mathrm{p}}-mN] \\ &\qquad e^{j2\pi\frac{nl_{\mathrm{p}}}{N}} e^{j2\pi\frac{(l-l_{\mathrm{p}}-mN)(k_{\mathrm{p}}+nM)}{MN}}, \end{aligned} \tag{2}$$

where $w[k,l] = \frac{1}{MN} e^{j2\pi\frac{q(k^2+l^2)}{MN}}$ is the MN-periodic discrete chirp filter with slope parameter q. This filtering operation distributes the pilot energy uniformly across the DD grid while maintaining unit average energy. Then, the discrete DD superimposed signal becomes

$$x_{\mathrm{dd}}[k,l] = \sqrt{E_{\mathrm{d}}}\, x_{\mathrm{d,dd}}[k,l] + \sqrt{E_{\mathrm{p}}}\, x_{\mathrm{s,dd}}[k,l], \tag{3}$$

where E_{d} and E_{p} denote the energy allocated to the data and pilot signals, respectively.

The $x_{\mathrm{dd}}[k,l]$s are supported on the information lattice $\Lambda_{\mathrm{dd}} = \{(k\frac{\tau_{\mathrm{p}}}{M}, l\frac{\nu_{\mathrm{p}}}{N})|k,l \in \mathbb{Z}\}$. The continuous DD superimposed signal is given by $x_{\mathrm{dd}}(\tau,\nu) = \sum_{k,l\in\mathbb{Z}} x_{\mathrm{dd}}[k,l]\delta(\tau - \frac{k\tau_{\mathrm{p}}}{M})\delta(\nu - \frac{l\nu_{\mathrm{p}}}{N})$, where $\delta(\cdot)$ denotes the continuous Dirac-delta function. A DD transmit pulse shaping filter $w_{\mathrm{tx}}(\tau,\nu)$ is then applied on $x_{\mathrm{dd}}(\tau,\nu)$ to limit the bandwidth and time duration of the transmitted signal. The DD transmitted signal becomes $x_{\mathrm{dd}}^{w_{\mathrm{tx}}}(\tau,\nu) = w_{\mathrm{tx}}(\tau,\nu) *_\sigma x_{\mathrm{dd}}(\tau,\nu)$, where $*_\sigma$ denotes the twisted convolution[2]. The transmitted time domain (TD) signal $s_{\mathrm{td}}(t)$ is the TD realization of $x_{\mathrm{dd}}^{w_{\mathrm{tx}}}(\tau,\nu)$, obtained by applying inverse Zak transform[3] as $s_{\mathrm{td}}(t) = \mathcal{Z}_t^{-1}(x_{\mathrm{dd}}^{w_{\mathrm{tx}}}(\tau,\nu))$. This propagates through a doubly-selective channel with DD impulse response $h_{\mathrm{phy}}(\tau,\nu) = \sum_{i=1}^{P} h_i \delta(\tau-\tau_i)\delta(\nu-\nu_i)$, where P denotes the number of DD paths, and the ith path has gain h_i, delay shift τ_i, and Doppler shift ν_i. The resulting TD output signal is denoted by $r_{\mathrm{td}}(t)$.

Received signal: The received TD signal at the receiver is given by $y_{\mathrm{td}}(t) = r_{\mathrm{td}}(t) + n_{\mathrm{td}}(t)$, where $n_{\mathrm{td}}(t)$ is AWGN with variance N_0. We define the data SNR, $\rho_{\mathrm{d}} = E_{\mathrm{d}}/MNN_0$ and the pilot SNR, $\rho_{\mathrm{p}} = E_{\mathrm{p}}/MNN_0$, yielding pilot to data power ratio, PDR $= \rho_{\mathrm{p}}/\rho_{\mathrm{d}} = E_{\mathrm{p}}/E_{\mathrm{d}}$. The received signal $y_{\mathrm{td}}(t)$ is converted to DD domain via Zak transform[4], yielding $y_{\mathrm{dd}}(\tau,\nu) = \mathcal{Z}_t(y_{\mathrm{td}}(t)) = r_{\mathrm{dd}}(\tau,\nu) + n_{\mathrm{dd}}(\tau,\nu)$, where $r_{\mathrm{dd}}(\tau,\nu) = h_{\mathrm{phy}}(\tau,\nu) *_\sigma w_{\mathrm{tx}}(\tau,\nu) *_\sigma x_{\mathrm{dd}}(\tau,\nu)$, and $n_{\mathrm{dd}}(\tau,\nu) = \mathcal{Z}_t(n_{\mathrm{td}}(t))$. The DD receive matched filter $w_{\mathrm{rx}}(\tau,\nu) = w_{\mathrm{tx}}^*(-\tau,-\nu)e^{j2\pi\nu\tau}$ is applied on $y_{\mathrm{dd}}(\tau,\nu)$ to give the filtered output

$$\begin{aligned} y_{\mathrm{dd}}^{w_{\mathrm{rx}}}(\tau,\nu) &= w_{\mathrm{rx}}(\tau,\nu) *_\sigma y_{\mathrm{dd}}(\tau,\nu) \\ &= \underbrace{w_{\mathrm{rx}}(\tau,\nu) *_\sigma h_{\mathrm{phy}}(\tau,\nu) *_\sigma w_{\mathrm{tx}}(\tau,\nu)}_{\triangleq\, h_{\mathrm{eff}}(\tau,\nu)} *_\sigma x_{\mathrm{dd}}(\tau,\nu) \\ &\quad + \underbrace{w_{\mathrm{rx}}(\tau,\nu) *_\sigma n_{\mathrm{dd}}(\tau,\nu)}_{\triangleq\, n_{\mathrm{dd}}^{w_{\mathrm{rx}}}(\tau,\nu)}, \end{aligned} \tag{4}$$

[1] $a[k,l] \circledast_{\sigma\mathrm{d}} b[k,l] \triangleq \sum_{k',l'=0}^{MN-1} a[k',l']b[k-k',l-l']e^{j2\pi\frac{l'(k-k')}{MN}}$.

[2] $a(\tau,\nu) *_\sigma b(\tau,\nu) \triangleq \iint a(\tau',\nu')b(\tau-\tau',\nu-\nu')e^{j2\pi\nu'(\tau-\tau')}d\tau' d\nu'$.

[3] $\mathcal{Z}_t^{-1}(a(\tau,\nu)) \triangleq \sqrt{\tau_{\mathrm{p}}}\int_0^{\nu_{\mathrm{p}}} a(t,\nu)d\nu$.

[4] $\mathcal{Z}_t\,(a(t)) \triangleq \sqrt{\tau_{\mathrm{p}}}\sum_{k\in\mathbb{Z}} a(\tau+k\tau_{\mathrm{p}})e^{-j2\pi\nu k\tau_{\mathrm{p}}}$.

where $h_{\text{eff}}(\tau,\nu)$ denotes the continuous effective channel filter consisting of the twisted convolution cascade of $w_{\text{tx}}(\tau,\nu)$, $h_{\text{phy}}(\tau,\nu)$, and $w_{\text{rx}}(\tau,\nu)$, and $n_{\text{dd}}^{w_{\text{rx}}}(\tau,\nu)$ denotes the noise filtered through the matched filter. Sampling $y_{\text{dd}}^{w_{\text{rx}}}(\tau,\nu)$ on the information lattice Λ_{dd} results in the discrete DD domain input-output relation

$$\begin{aligned} y_{\text{dd}}[k,l] &= h_{\text{eff}}[k,l] *_{\sigma\text{d}} x_{\text{dd}}[k,l] + n_{\text{dd}}[k,l] \\ &= \sum_{k',l'\in\mathbb{Z}} h_{\text{eff}}[k-k',l-l'] x_{\text{dd}}[k',l'] e^{j2\pi\frac{k'(l-l')}{MN}} + n_{\text{dd}}[k,l]. \end{aligned} \quad (5)$$

Here, $*_{\sigma\text{d}}$ denotes discrete twisted convolution, the discrete effective channel filter $h_{\text{eff}}[k,l] = h_{\text{eff}}(\tau=\frac{k\tau_{\text{p}}}{M},\nu=\frac{l\nu_{\text{p}}}{N})$, and the filtered noise samples $n_{\text{dd}}[k,l] = n_{\text{dd}}^{w_{\text{rx}}}(\tau=\frac{k\tau_{\text{p}}}{M},\nu=\frac{l\nu_{\text{p}}}{N})$.

Matrix-vector form: Leveraging the quasi-periodicity in the DD domain, it is sufficient to consider the received samples $y_{\text{dd}}[k,l]$ within $\mathcal{D}_0$. Then, the received signal model can be written in matrix-vector form as:

$$\mathbf{y} = \mathbf{H}_{\text{eff}}\mathbf{x} + \mathbf{n}, \quad (6)$$

where $\mathbf{x},\mathbf{y},\mathbf{n} \in \mathbb{C}^{MN\times 1}$, such that their $(kN+l+1)$th entries are given by $x_{kN+l+1} = x_{\text{dd}}[k,l]$, $y_{kN+l+1} = y_{\text{dd}}[k,l]$, $n_{kN+l+1} = n_{\text{dd}}[k,l]$ and $\mathbf{H}_{\text{eff}} \in \mathbb{C}^{MN\times MN}$ is the effective channel matrix such that

$$\begin{aligned} \mathbf{H}_{\text{eff}}[k'N+l'+1, kN+l+1] = \sum_{m,n\in\mathbb{Z}} h_{\text{eff}}[k'-k-nM, \\ l'-l-mN] e^{j2\pi nl/N} e^{j2\pi\frac{(l'-l-mN)(k+nM)}{MN}}, \end{aligned} \quad (7)$$

for $k',k \in \mathbb{K}$, and $l',l \in \mathbb{L}$.

III. Receiver processing techniques

To address the pilot-data interference inherent in SSP frames, we investigate four receiver schemes. The common objective across all the schemes is to extract the effective channel estimate $\hat{h}_{\text{eff}}[k,l]$ and subsequently detect the data symbols $\hat{x}[k,l]$.

A. Baseline read-off estimator

This approach [10] exploits the properties of the self-ambiguity function of the spread pilot, which is supported on a twisted lattice Λ_q rotated relative to the period lattice on which the data signal is supported. The receiver computes the cross-ambiguity function (CAF) between the received signal $y_{\text{dd}}[k,l]$ and the spread pilot signal $x_{\text{s,dd}}[k,l]$:

$$A_{y,x_s}[k,l] = \sum_{k'=0}^{M-1}\sum_{l'=0}^{N-1} y_{\text{dd}}[k',l'] x^*_{\text{s,dd}}[k'-k,l'-l] e^{-j2\pi\frac{l(k'-k)}{MN}}. \quad (8)$$

This CAF output comprises the desired effective channel response superimposed with noise and interference (DD aliasing and data interference) whose severity is governed by the pulse shape characteristics. The baseline estimator ignores these interference-plus-noise components and simply "reads off" the CAF within a support set $\mathcal{S}$ as

$$\hat{h}_{\text{eff}}[k,l] = \frac{A_{y,x_s}[k,l]}{\sqrt{E_{\text{p}}}}, \quad \text{for } (k,l)\in\mathcal{S}. \quad (9)$$

In our simulations, we define $\mathcal{S}$ geometrically as a rhombus centered at the origin, with diagonal lengths extending 16 and 20 grid points along the delay and Dopper axes, respectively. This compact support set is specifically chosen to tightly encompass the effective channel energy while excluding distant grid points dominated by noise and aliasing. To recover the data, the pilot contribution is reconstructed using this estimate and subtracted from the received signal to obtain the data-only component: $y_{\text{dd}}[k,l] - \sqrt{E_{\text{p}}}(\hat{h}_{\text{eff}} *_{\sigma\text{d}} x_{\text{s,dd}})[k,l]$, on which minimum mean square error (MMSE) detection is performed to obtain the data symbol estimates $\hat{x}[k,l]$.

B. Turbo receiver

To mitigate the effect of data interference that limits the baseline estimator performance, the turbo receiver [11] employs an iterative interference cancellation loop initialized with the baseline estimates. This architecture alternates between channel estimation and data detection to progressively refine the signal quality. The turbo process proceeds as follows for the tth iteration.

Step 1: Data interference cancellation: Using the estimates from the previous iteration ($\hat{h}_{\text{eff}}^{(t-1)}$ and $\hat{x}^{(t-1)}$), the receiver reconstructs the data component and subtracts it from the received signal to isolate the pilot contribution as

$$y_{\text{s,dd}}^{(t)}[k,l] = y_{\text{dd}}[k,l] - \sqrt{E_{\text{d}}}(\hat{h}_{\text{eff}}^{(t-1)} *_{\sigma\text{d}} \hat{x}_{\text{d,dd}}^{(t-1)})[k,l], \quad (10)$$

where $\hat{x}_{\text{d,dd}}^{(t-1)}$ is the discrete DD data signal reconstructed from the detected symbols $\hat{x}^{(t-1)}$.

Step 2: Channel estimation update: The effective channel estimate is updated by computing the CAF on the cleaned pilot signal $y_{\text{s,dd}}^{(t)}$ as

$$\hat{h}_{\text{eff}}^{(t)}[k,l] = \frac{A_{y_{\text{s}}^{(t)},x_s}[k,l]}{\sqrt{E_{\text{p}}}}, \quad \text{for } (k,l)\in\mathcal{S}. \quad (11)$$

Step 3: Pilot interference cancellation: The updated channel estimate is used to reconstruct the pilot component, which is then subtracted from the original received signal to yield a cleaner data signal as

$$y_{\text{d,dd}}^{(t)}[k,l] = y_{\text{dd}}[k,l] - \sqrt{E_{\text{p}}}(\hat{h}_{\text{eff}}^{(t)} *_{\sigma\text{d}} x_{\text{s,dd}})[k,l]. \quad (12)$$

Step 4: Data detection update: Finally, MMSE detection is performed on $y_{\text{d,dd}}^{(t)}$ to obtain the refined data symbols $\hat{x}^{(t)}[k,l]$ for the next iteration.

This loop continues for a fixed number of iterations. While effective, the performance of the turbo scheme is sensitive to error propagation. Poor initial estimates from the baseline readoff can lead to erroneous data cancellation, causing the loop to diverge.

C. CNN denoiser

In this scheme [12], a CNN-based estimation framework formulates the channel estimation problem as a 2D image denoising task. The noisy CAF output $A_{y,x_s}[k,l] \in \mathcal{S}_c$ is modeled as a noisy image corrupted by DD aliasing, data interference, and AWGN. While the baseline estimator (Sec.

III-A) has a tight rhombus support $\mathcal{S}$ to minimize noise inclusion, the CNN employs an extended rectangular support set $\mathcal{S}_c = \left\{(k,l) : -\lfloor \frac{M_c}{2} \rfloor \leq k \leq \lfloor \frac{M_c}{2} \rfloor, -\lfloor \frac{N_c}{2} \rfloor \leq l \leq \lfloor \frac{N_c}{2} \rfloor\right\}$ centered at the origin. In our simulations, we set $M_c = 27$ and $N_c = 43$. This wider region captures the full channel context, including surrounding interference patterns, allowing the CNN to distinguish and learn signal from noise.

1) Data representation: We represent the ground truth and noisy input images as complex-valued matrices $\mathbf{I}_{\mathrm{h}}^{\mathrm{true}}, \mathbf{I}_{\mathrm{h}}^{\mathrm{noisy}} \in \mathbb{C}^{M_c \times N_c}$. The mapping from the DD grid to the matrix entries is defined as $\mathbf{I}_{\mathrm{h}}^{\mathrm{true}}[m,n] = h_{\mathrm{eff}}\left[m - \Delta_k, n - \Delta_l\right]$ and $\mathbf{I}_{\mathrm{h}}^{\mathrm{noisy}}[m,n] = A_{y,x_s}\left[m - \Delta_k, n - \Delta_l\right] / \sqrt{E_{\mathrm{p}}}$, where $m \in \{0,\ldots,M_c-1\}$ and $n \in \{0,\ldots,N_c-1\}$ are the matrix indices, and the shifts are given by $\Delta_k = \lfloor \frac{M_c}{2} \rfloor$ and $\Delta_l = \lfloor \frac{N_c}{2} \rfloor$. Since standard CNNs operate on real values, we decompose these complex images into their real and imaginary components [12]. The network processes these components $\Re\{\mathbf{I}_{\mathrm{h}}^{\mathrm{noisy}}\}$ and $\Im\{\mathbf{I}_{\mathrm{h}}^{\mathrm{noisy}}\}$ independently and gives the final complex-valued enhanced image as $\mathbf{I}_{\mathrm{h}}^{\mathrm{CNN}} = \mathcal{F}_{\mathrm{CNN}}(\Re\{\mathbf{I}_{\mathrm{h}}^{\mathrm{noisy}}\}) + j\mathcal{F}_{\mathrm{CNN}}(\Im\{\mathbf{I}_{\mathrm{h}}^{\mathrm{noisy}}\})$, where $\mathcal{F}_{\mathrm{CNN}}(\cdot)$ denotes the CNN denoising function. The final channel estimate is obtained by mapping the matrix entries back to the support $\mathcal{S}_c$ as

$$\hat{h}_{\mathrm{CNN}}[k,l] = \mathbf{I}_{\mathrm{h}}^{\mathrm{CNN}}[k + \Delta_k, l + \Delta_l], \quad \text{for } (k,l) \in \mathcal{S}_c. \tag{13}$$

2) Network architecture: We use a four-layer hierarchical network tailored to the specific dimensions of the DD grid and the support set $\mathcal{S}_c$ detailed in [12]:

- ***Layer 1 (Global feature extraction):*** A layer with 64 filters with a large kernel size 27×27 and rectified linear unit (ReLU) activation. This large receptive field is useful for capturing global DD correlations especially characteristic of poorly localized pulses like sinc.
- ***Layer 2 (Feature compression):*** A bottleneck layer with 32 filters of kernel size 9×9 and ReLU activation reduces the dimensionality while preserving the spatial context of the extracted features.
- ***Layer 3 (Feature refinement):*** A layer with 32 filters of size 5×5 and ReLU activation to refine the features.
- ***Layer 4 (Reconstruction):*** A final layer with a single 15×15 filter and linear activation that aggregates the refined features to reconstruct the denoised channel image.

This architecture contains 245,473 trainable parameters.

3) Robust training methodology: To ensure the receiver is robust across different interference regimes, the network is not trained on a single operating point. Instead, we generate a diverse training dataset at a fixed data SNR of 15 dB but mixed across multiple PDRs of $\{0, 5, 15, 25\}$ dB, for each pulse shaping filter. The CNN is implemented and trained using the MATLAB Deep Learning Toolbox with GPU acceleration on an NVIDIA TITAN RTX platform. The network parameters are optimized using the Adam optimizer to minimize the normalized mean square error (NMSE) loss function:

$$\mathcal{L} = \frac{1}{N_{\mathrm{s}}} \sum_{i=1}^{N_{\mathrm{s}}} \frac{\|\mathbf{I}_i^{\mathrm{true}} - \mathbf{I}_i^{\mathrm{CNN}}\|_F^2}{\|\mathbf{I}_i^{\mathrm{true}}\|_F^2}, \tag{14}$$

TABLE I: CNN training parameters

Parameter	**Value**
Dataset specifications (per pulse)	
Training data SNR	15 dB
Training PDRs	$\{0, 5, 15, 25\}$ dB
Samples per PDR	80,000
Total training inputs	640,000 (Real/Imag split)
Optimization settings	
Optimizer	Adam
Initial learning rate	5×10^{-4}
Learning rate schedule	Piecewise (Factor 0.8, 5 epochs)
Mini-batch size	64
Max epochs	50

TABLE II: Computational complexity comparison

Receiver	**Dominant Term**	**Total Complexity Order**
Baseline	MMSE	$\mathcal{O}(M^3N^3)$
Turbo	MMSE loop	$(Q_{\mathrm{iter}}+1) \cdot \mathcal{O}(M^3N^3)$
CNN	MMSE + CNN	$\mathcal{O}(M^3N^3) + C_{\mathrm{net}}$
Hybrid	MMSE + CNN loop	$(Q_{\mathrm{hyb}}+1) \cdot [\mathcal{O}(M^3N^3) + C_{\mathrm{net}}]$

where N_{s} denotes the total number of training samples, and $\mathbf{I}_i^{\mathrm{true}}$ and $\mathbf{I}_i^{\mathrm{CNN}}$ represent the ground truth image (label) and the denoised network output for the ith sample, respectively. The specific hyperparameters and dataset specifications are summarized in Table I.

D. Proposed hybrid receiver (CNN + turbo)

We propose a hybrid architecture that combines the robustness of deep learning with the precision of iterative interference cancellation. The motivation is twofold: the CNN effectively suppresses the interference and noise that linear readoff cannot address and gives more accurate channel estimates, while the turbo loop iteratively cancels the interference to refine the signal quality. In this proposed scheme, the learning-based denoiser is integrated directly into the turbo loop. Specifically, in each iteration t (starting from the initialization $t = 0$), the raw channel estimate obtained from the CAF readoff is processed by the pre-trained CNN to yield a high-quality estimate in (13). This creates an optimal trade-off: the CNN lowers the channel estimation error floor at every step, ensuring that the MMSE detector operates in a high-SINR regime. This mechanism is critical for preventing error propagation, particularly when employing poorly localized pulses (e.g., sinc). In such cases, the severe DD aliasing renders standard linear estimates too poor for the conventional turbo loop to converge.

E. Computational complexity analysis

Table II summarizes the computational complexity of the considered receivers in terms of the number of real-valued floating-point operations (FLOPs) per frame.

- ***Baseline:*** The complexity is dominated by the MMSE detection, scaling cubically as $\mathcal{O}(M^3N^3)$. The combined cost of CAF-based estimation, channel matrix construction, and pilot interference cancellation $(14MN|\mathcal{S}| +$

$78M^2N^2$) is small in comparison. The total complexity is: $C_{\text{Base}} \approx \mathcal{O}(M^3N^3) + 78M^2N^2 + 14MN|\mathcal{S}|$.

- ***Turbo:*** This scheme performs an initial baseline stage followed by Q_{iter} iterative updates. Each iteration repeats the baseline operations and adds a data interference cancellation cost of $8M^2N^2$. The total complexity is $C_{\text{Turbo}} \approx (Q_{\text{iter}} + 1)C_{\text{Base}} + Q_{\text{iter}}(8M^2N^2)$.
- ***CNN denoiser:*** This scheme modifies the baseline by using an extended rectangular support $\mathcal{S}_c$ (size $M_c \times N_c$) and inserting a CNN denoising module. The total complexity is $C_{\text{CNN}} \approx \mathcal{O}(M^3N^3)+78M^2N^2+14MN|\mathcal{S}_c|+C_{\text{net}}$. Since training is offline, the term C_{net} represents only the forward inference pass. It scales linearly with the support size $C_{\text{net}} \approx 2 \cdot (M_cN_c) \cdot \Omega_{\text{net}}$, where $\Omega_{\text{net}} \approx 4.9 \times 10^5$ FLOPs/pixel denotes the aggregate kernel operations per pixel across all layers for the architecture used.
- ***Proposed hybrid:*** Integrating the CNN denoiser into the turbo loop for Q_{hyb} iterations results in a total complexity of $C_{\text{Hybrid}} \approx (Q_{\text{hyb}} + 1)C_{\text{CNN}} + Q_{\text{hyb}}(8M^2N^2)$.

IV. Results and discussions

This section presents simulation results on the channel estimation and data detection performance of Zak-OTFS with SSP under different receiver processing techniques and pulse shaping filters. The performance is evaluated in terms of NMSE of the effective channel estimate and bit error rate (BER) of data detection. Sinc, Gaussian, and GS filters are considered. Vehicular-A channel model [13] having $P = 6$ paths with fractional DDs and a power delay profile given in Table 2 of [2] with a maximum delay spread of 2.51 μs is used. The maximum Doppler shift is $\nu_{\max} = 815$ Hz, and the Doppler shift of the ith path is modeled as $\nu_i = \nu_{\max}\cos\theta_i, i = 1, \ldots, P$, where θ_is are independent and uniformly distributed in $[0, 2\pi)$. A system configuration with frame size $M = 31$ and $N = 37$ is used. The slope parameter of the discrete chirp spreading filter is set to $q = 3$. A Doppler period of $\nu_p = 30$ kHz is considered, with a resulting bandwidth of $B = M\nu_p = 930$ kHz and frame duration $T = N\tau_p = 1.23$ ms. The considered system parameters satisfy the crystallization condition. Also, in the simulations, the range of values of m and n in (7) is limited to -1 to 1, and this is found to ensure an adequate support set of $h_{\text{eff}}[k, l]$ that captures the channel spread accurately. BPSK modulation and MMSE detection are used. For each pulse shape, the operating PDR is chosen to minimize the bit error rate (BER) in the corresponding BER vs PDR plot, and subsequent results are presented at this operating point.

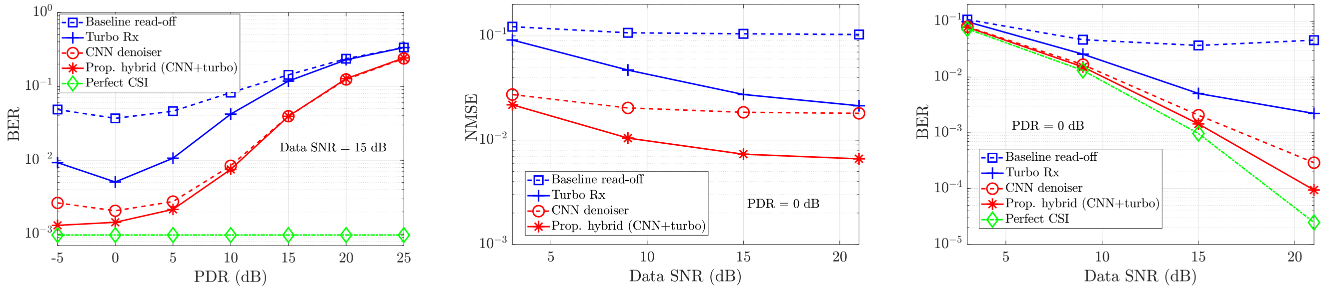
Performance with sinc filter: The performance of the orthogonal but poorly localized sinc pulse is analyzed in Fig. 2. While its orthogonality (zero ISI at the DD sampling points) aids good data detection performance, the high-energy sidelobes cause widespread pilot-data interference and significant aliasing, degrading channel estimation performance. The BER versus PDR performance (Fig. 2a) highlights performance degradation at low PDRs due to poor estimation and at high PDRs due to excessive pilot interference, necessitating an optimized operating point of 0 dB. Even at this optimum, the baseline estimator is ineffective, as it exhibits a high error floor in both NMSE (Fig. 2b) and BER (Fig. 2c). The standard turbo receiver yields only marginal improvements as the initial linear estimate is too corrupted for the turbo loop to converge. While the CNN denoiser offers improvement, only the proposed hybrid receiver achieves very good detection performance (by using the CNN to get better estimates, enabling the turbo loop to effectively cancel residual interference) and approaches close to the perfect CSI benchmark.

Performance with Gaussian filter: In contrast to the sinc pulse, the Gaussian pulse exhibits optimal localization, significantly reducing pilot-data interference and aliasing, at the cost of non-orthogonality and severe ISI at DD sampling points. These characteristics are reflected in Fig. 3. The BER vs PDR curve (Fig. 3a) shows a broad and shallow minimum around low PDR values, indicating robustness to pilot power variations. While the baseline estimator achieves reasonable NMSE (Fig. 3b) due to localization, the high ISI causes severe BER degradation (Fig. 3c) that the standard turbo receiver fails to mitigate. Crucially, the CNN-based receiver effectively suppresses the residual non-orthogonal interference, achieving performance closely approaching the perfect CSI benchmark. Consequently, additional gains from the turbo or hybrid loops are marginal, demonstrating that for well-localized pulses (e.g., Gaussian), learning-based denoising is sufficient without the need for iterative interference cancellation.

Performance with GS filter: The GS pulse represents an intermediate regime, effectively combining the complementary strengths of the sinc and Gaussian pulses [8]. This balance is illustrated in Fig. 4. By improving localization relative to the sinc pulse, the GS filter significantly mitigates the widespread pilot-data interference and aliasing, while retaining orthogonality to keep ISI manageable. Consequently, the baseline readoff estimator provides a sufficiently accurate initialization, allowing the standard turbo receiver to converge effectively, contrary to the sinc scenario where it diverged. While the proposed hybrid receiver still achieves the best performance, closely matching the perfect CSI benchmark, the performance gap between the turbo and hybrid architectures is considerably narrower here. This confirms that optimizing the pulse shape to reduce interference structure (e.g., as in GS pulse) relaxes the dependency on the learning-aided estimation, making the standard turbo loop a viable low-complexity alternative.

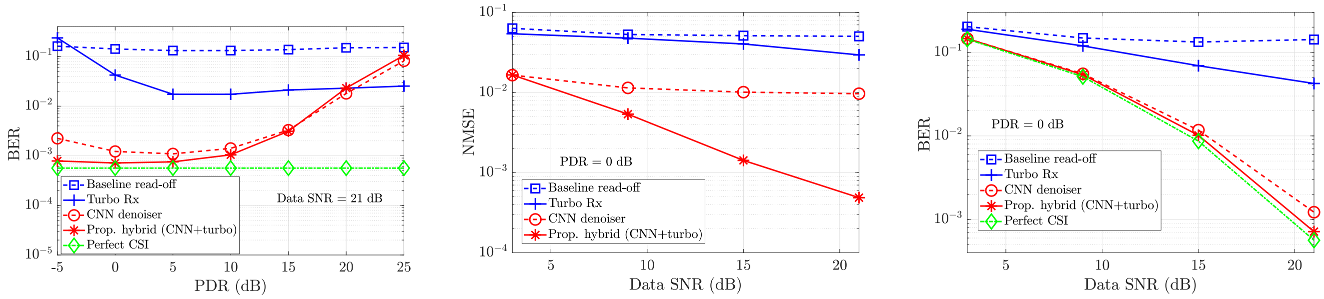
V. Conclusions

We investigated receiver techniques for Zak-OTFS with superimposed spread pilot. We demonstrated that the DD domain pulse shaping filter fundamentally determines the severity and structure of interference induced by superimposed spread pilot, and, as a result, the receiver signal processing required for reliable operation strongly depends on the pulse shape. Our results showed that the GS pulse, benefiting from both orthogonality and localization, is robust enough to perform well with a standard turbo receiver. The Gaussian pulse, despite its excellent localization, requires learning-based denoising to



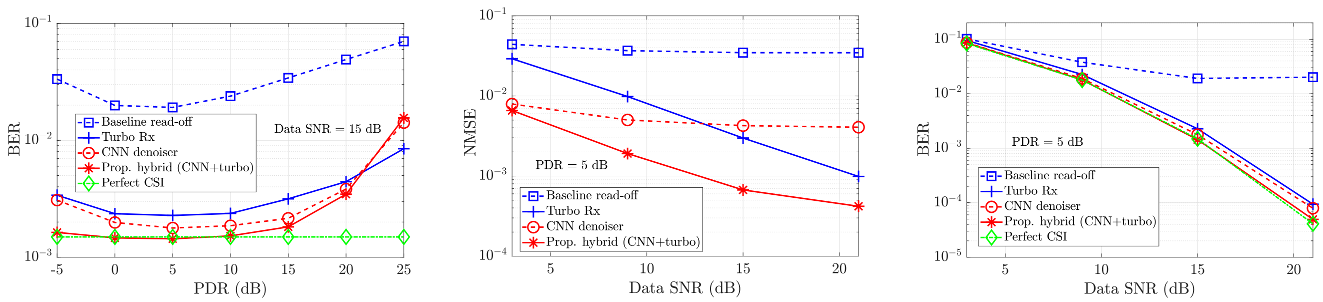
(a) BER vs PDR (b) NMSE vs data SNR (c) BER vs data SNR

Fig. 2: Performance of Zak-OTFS with superimposed spread pilot and sinc filter.



(a) BER vs PDR (b) NMSE vs data SNR (c) BER vs data SNR

Fig. 3: Performance of Zak-OTFS with superimposed spread pilot and Gaussian filter.



(a) BER vs PDR (b) NMSE vs data SNR (c) BER vs data SNR

Fig. 4: Performance of Zak-OTFS with superimposed spread pilot and Gaussian-sinc filter.

mitigate the effects of its non-orthogonality. In contrast, the sinc pulse, lacking localization despite being orthogonal, requires the proposed hybrid architecture to achieve performance close to its perfect CSI benchmark.

REFERENCES

[1] S. K. Mohammed, R. Hadani, A. Chockalingam, and R. Calderbank, "OTFS — a mathematical foundation for communication and radar sensing in the delay-Doppler domain," *IEEE BITS the Inform. Theory Mag.*, vol. 2, no. 2, pp. 36-55, 1 Nov. 2022.

[2] S. K. Mohammed, R. Hadani, A. Chockalingam, and R. Calderbank, "OTFS — predictability in the delay-Doppler domain and its value to communication and radar sensing," *IEEE BITS the Inform. Theory Mag.*, vol. 3, no. 2, pp. 7-31, Jun. 2023.

[3] S. K. Mohammed, R. Hadani, and A. Chockalingam, *OTFS Modulation – Theory and Applications*, IEEE Press-Wiley, 2024.

[4] S. Gopalam et al., "Zak-OTFS implementation via time and frequency windowing," *IEEE Trans. Commun.*, vol. 72, pp. 3873-3889, Jul. 2024.

[5] S. Gopalam, H. Inaltekin, I. B. Collings, and S. V. Hanly, "Optimal Zak-OTFS receiver and its relation to the radar matched filter," *IEEE Open J. of the Commun. Soc.*, vol. 5, pp. 4462-4482, 2024.

[6] S. Li, W. Yuan, Z. Wei, J. Yuan, B. Bai, and G. Caire, "On the pulse shaping for delay-Doppler communications," *Proc. IEEE GLOBECOM'2024*, pp. 4909-4914, Dec. 2023.

[7] J. Jayachandran et al., "Zak-OTFS: pulse shaping and the tradeoff between time/bandwidth expansion and predictability," online arxiv.org/abs/2405.02718, 4 May 2024.

[8] A. Das, F. Jesbin, and A. Chockalingam, "A Gaussian-sinc pulse shaping filter for Zak-OTFS," *IEEE Trans. Veh. Tech.*, doi: 10.1109/TVT.2025.3609781, 2025.

[9] J. Jayachandran et al., "Zak-OTFS with interleaved pilots to extend the region of predictable operation," *IEEE Trans. Veh. Tech.*, vol. 74, no. 11, pp. 17545-17559, Nov. 2025.

[10] M. Ubadah et al., "Zak-OTFS to integrate sensing the I/O relation and data communication," online: arXiv:2404.04182, Jun. 2025.

[11] J. Jayachandran, M. Ubadah, S. K. Mohammed, R. Hadani, and A. Chockalingam, "Zak-OTFS and turbo signal processing for joint sensing and communication," online: arXiv:2406.06024, 10 Jun. 2024.

[12] C. D. Kabade, A. Das and A. Chockalingam, "Zak-OTFS with superimposed spread pilot: CNN-aided channel estimation," *Proc. IEEE PIMRC'2025*, pp. 1-6, Sep. 2025.

[13] ITU-R M.1225, "Guidelines for evaluation of radio transmission technologies for IMT-2000," Int. Telecom. Union Radio Commun., 1997.