Cross-Layer Design for Near-Field mmWave Beam Management and Scheduling under Delay-Sensitive Traffic

Zijun Wang
Department of Electrical Engineering
University at Buffalo
Buffalo, NY 14226 USA
zwang267@buffalo.edu

Anjali Omer
Department of Electrical Engineering
University at Buffalo
Buffalo, NY 14226 USA
anjaliom@buffalo.edu

Jacob Chakareski
Ying Wu College of Computing
New Jersey Institute for Technology
Newark, NJ 07103 USA
jacobcha@njit.edu

Nicholas Mastronarde
Department of Electrical Engineering
University at Buffalo
Buffalo, NY 14226 USA
nmastron@buffalo.edu

Rui Zhang
Department of Electrical Engineering
University at Buffalo
Buffalo, NY 14226 USA
rzhang45@buffalo.edu

Abstract

Next-generation wireless networks will rely on mmWave/sub-THz spectrum and extremely large antenna arrays (ELAAs). This will push their operation into the near-field where far-field beam management degrades and beam training becomes more costly and must be done more frequently. Because ELAA training and data transmission consume energy and training trades off with service time, we pose a cross-layer control problem that couples PHY-layer beam management with MAC-layer service under delay-sensitive traffic. The controller decides when to retrain and how aggressively to train (pilot count and sparsity) while allocating transmit power, explicitly balancing pilot overhead, data-phase rate, and energy to reduce the queueing delay of MAC-layer frames/packets to be transmitted. We model the problem as a partially observable Markov decision process and solve it with deep reinforcement learning. In simulations with a realistic near-field channel and varying mobility and traffic load, the learned policy outperforms strong 5G-NR–style baselines at a comparable energy: it achieves 85.5% higher throughput than DFT sweeping and reduces the overflow rate by 78%. These results indicate a practical path to overhead-aware, traffic-adaptive near-field beam management with implications for emerging low-latency high-rate next-generation applications such as digital twin, spatial computing, and immersive communication.

1 Introduction

Millimeter-wave (mmWave) and sub-THz bands offer abundant spectrum for high-speed links, while extremely large antenna arrays (ELAAs) are employed to overcome the associated high path loss (1; 2; 3; 4). However, the use of ELAAs means that some uses, traditionally assumed to operate

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: AI and ML for Next-Generation Wireless Communications and Networking (AI4NextG).

in the far-field regime, now fall within the near-field region (5; 6; 7; 8). Conventional far-field design models electromagnetic waves as planar (9). Under this assumption, array responses vary primarily by angle-dependent phase shifts, enabling the channel to be compactly represented using a discrete Fourier transform (DFT) basis, with sparsity roughly corresponding to the number of dominant paths (10). In the near field, wavefronts are spherical, and the steering vector becomes a function of both angle and range. As a result, the channel's DFT-domain representation is no longer sparse in a way that simply reflects the number of paths; instead, its sparsity varies with user geometry and mobility (11; 12). This undermines the use of far-field codebooks and increases the risk of misalignment. Near-field codebooks have been explored (13), yet larger dictionaries and codeword correlation increase training time and energy overhead that directly competes with data transmission. A pragmatic alternative is to combine 5G-new radio (NR)-style sweeping (14) with near-field codebooks to avoid explicit sparsity selection (15); still, the sweeping/reporting overhead can exceed that of standard NR due to the expanded search space.

Critically, beam training is deeply intertwined with MAC-layer service scheduling (16). Each pilot transmission consumes airtime and energy that could otherwise be used to serve queued traffic. Under bursty arrivals, finite buffers, and mobility-induced dynamics, a PHY-only design that optimizes instantaneous link metrics can still hurt end-to-end latency. This motivates a cross-layer treatment that co-optimizes training timing and intensity with power allocation and queue-aware scheduling (17; 18).

Prior work has partially addressed this coupling in directional and mmWave systems. Shokri-Ghadikolaei et al. formalize the alignment–throughput trade-off and propose joint beamwidth/scheduling strategies (19), while subsequent work coordinates beam schedules with mobility and sleep/wake cycles to target energy, delay, or throughput (20; 21; 22). Lei et al. show that adaptive retraining and power control can significantly improve delay/energy compared with fixed policies (16). However, these studies primarily assume far-field propagation and do not tackle near-field-specific issues: angle–range coupling, expanded codebooks, and variable sparsity that governs training intensity.

This paper formulates near-field beam management as a cross-layer control problem for delay-sensitive traffic with minimal energy consumption. We propose a queue-aware policy that jointly decides when to retrain and how aggressively to train (pilot budget and sparsity level), together with data-phase power allocation. Our implementation incorporates compressive-sensing-based training and a deep reinforcement learning (DRL)-based controller that observes queue states and recent training history to balance pilot overhead, service rate, and energy. In simulations with near-field channels over a range of mobility and load models, the learned policy reduces queueing delay and overflow at a comparable energy to strong baselines. The proposed approach narrows the gap to full-channel state information (CSI) performance while offering an overhead-aware and traffic-adaptive solution. Our advances can have implications for emerging low-latency high-rate next-generation applications such as digital twin, spatial computing, and immersive communication that increasingly integrate mmWave capabilities (23; 24; 25; 26; 27).

2 System Model

In this section, we discuss the channel model, beam training method and data queuing model, which are essential for understanding our cross-layer decision model presented in Section 3.

2.1 Channel Model

We consider a narrow band multiple-input single-output (MISO) mmWave communication system as shown in Fig. 1, where the gNB is equipped with a uniform planar array (UPA). The UPA is placed on the $x-z$ plane and the center of the UPA is at $\mathbf{0} = (0, 0, 0)$. The number of antenna elements of the UPA is $M = M_z \times M_x$, where M_z and M_x are the number of elements in the z and x directions, respectively. For a UPA, the near-field region lies between the Fresnel distance $R_{Fre} = \frac{1}{2}\sqrt{\frac{D^3}{\lambda}}$ and the Rayleigh distance $R_{Ray} = \frac{2D^2}{\lambda}$, where λ is the wavelength at the central frequency, $D = \sqrt{(M_x d)^2 + (M_z d)^2}$ is the aperture of the UPA (28), and $d = \frac{\lambda}{2}$ is the spacing of the antenna elements. The far field lies past R_{Ray}. In this paper, we focus on cross-layer beam management and data transmission of users in the near field region between R_{Fre} and R_{Ray}.

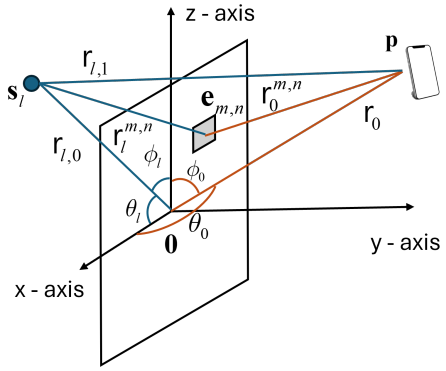

Figure 1: Near-field communication system with UPA.

We consider a multi-path ray-tracing channel model as shown in Fig. 1. Suppose that the user is positioned at $\mathbf{p}$, and the distance from the center of the antenna to the user is $r_0 = \|\mathbf{p}\|$. Then the distance $r_0^{m,n}$ between the (m,n)-th antenna element with coordinate $\mathbf{e}_{m,n}$ and the user is

$$r_0^{m,n} = \|\mathbf{p} - \mathbf{e}_{m,n}\| = \sqrt{r_0^2 + \delta_n^2 d^2 + \delta_m^2 d^2 - 2r_0\delta_n \cos\theta_0 \sin\phi_0\, d - 2r_0\delta_m \cos\phi_0\, d}\,, \tag{1}$$

where

$$\delta_n = n - \frac{M_x+1}{2},\ \delta_m = m - \frac{M_z+1}{2},\ \mathbf{e}_{m,n} = \begin{bmatrix} \delta_n d \\ 0 \\ \delta_m d \end{bmatrix},\ m = 1,\ldots,M_z,\ n = 1,\ldots,M_x.$$

We use the exact per-element phase but approximate the amplitude by the center distance to get a point to point (P2P) line-of sight (LoS) channel model from the (m,n)-th antenna element to the user:

$$g_{m,n}^{(0)} \approx \frac{4\pi}{\lambda r_0} \exp\Big(-j\frac{2\pi}{\lambda} r_0^{m,n}\Big).$$

Suppose $\mathbf{s}_\ell$ is the coordinate of the ℓ-th scatterer and the distance from this scatter to the (m,n)-th antenna element is $r_\ell^{m,n}$, then a similar calculation can be done as in Eq. (1).

Define

$$r_{\ell,0} = \|\mathbf{s}_\ell\| \ \text{(array-center-to-scatterer distance)} \ \text{ and } \ r_{\ell,1} = \|\mathbf{p} - \mathbf{s}_\ell\| \ \text{(scatterer-to-user distance)}.$$

Then for the non-line-of-sight (NLoS) path, the $r_{\ell,1}$ path will introduce extra path loss and phase delay compared to the LoS path:

$$g_{m,n}^{(\ell)} \approx \frac{4\pi}{\lambda r_{\ell,0}\, r_{\ell,1}} \exp\Big(-j\frac{2\pi}{\lambda}(r_\ell^{m,n} + r_{\ell,1})\Big).$$

Collecting paths and optionally absorbing the approximate amplitude into path coefficients yields the convenient representation $\mathbf{h} = \sum_{\ell=0}^{L-1} \beta_\ell\, \tilde{\mathbf{g}}^{(\ell)}$, where $\tilde{g}_{m,n}^{(\ell)} = \exp\Big(-j\frac{2\pi}{\lambda} r_\ell^{m,n}\Big)$, and the scalar path gain β_ℓ is chosen as $\beta_0 = \frac{4\pi}{\lambda r_0},\ \beta_\ell = \frac{4\pi}{\lambda r_{\ell,0}\, r_{\ell,1}} \exp\Big(-j\frac{2\pi}{\lambda} r_{\ell,1}\Big) (\ell \geq 1)$.

2.2 Beam Training based on Compressive Sensing

We introduce compressive-sensing-based beam training and the time division for beam training and data transmission in one frame in this section. Suppose the channel can be sparsely represented by a DFT basis. Then, for a DFT codebook matrix $\mathbf{F}$, we have

$$\mathbf{h} = \mathbf{F}\alpha, \tag{2}$$

where the implementation of the DFT matrix can be found in Section A. In our implementation, we take the DFT grid sizes equal to the UPA dimensions (i.e., $\mathbf{F} \in \mathbb{C}^{M\times M}$) and the sensing matrix Φ is realized as a Gaussian mixing applied on the DFT codebook matrix. Concretely, let

$$\Phi = \mathbf{F}\, G^H \in \mathbb{C}^{M\times m}, \ \text{ where } \ G \in \mathbb{C}^{m\times M} \ \text{ and } \ G_{ij} \sim \mathcal{CN}(0, 1/m).$$

For a pilot vector $\mathbf{x} \in \mathbb{C}^m$ the scalar observation is written as (using Eq. (2))

$$y \;=\; \mathbf{h}^H \Phi \mathbf{x} + w \;=\; (\mathbf{F}\alpha)^H \Phi \mathbf{x} + w \;=\; \alpha^H \mathbf{A}\mathbf{x} + w,$$

where $w \sim \mathcal{CN}(0, \sigma)$ is additive gaussian noise with power σ^2. Specializing to canonical per-pilot transmissions $\mathbf{x} = \mathbf{e}_i$ yields $y_i = \alpha^H \mathbf{A}\mathbf{e}_i + w_i = \alpha^H \boldsymbol{a}_i + w_i$ with $\boldsymbol{a}_i$ the i-th column of $\mathbf{A}$. Stacking the m measurements as a row vector gives the compact row-form $\mathbf{y} \;=\; \alpha^H \mathbf{A} + \mathbf{w}, \;\; \mathbf{y} \in \mathbb{C}^{1\times m}$.

We use a revised Target-sparsity Subspace Pursuit (TSP) for compressive sensing (29) which takes $\mathbf{y}$, $\mathbf{A}$ and sparsity level k as input and outputs the recovered coefficients $\hat{\alpha}$. The detailed algorithm can be found in Section C. According to compressive sensing theory, a sufficient condition for exact recovery is that the number of pilot measurements m scales with the sparsity level k (30). Because the sparsity of a near-field channel in a DFT dictionary is not fixed and can vary over time and geometry, we let a DRL controller adaptively choose both the sensing dimension (pilot budget m) and the working sparsity level k online, rather than fixing them a priori.

From the recovered coefficients $\hat{\alpha}$ we form the channel estimate $\hat{\mathbf{h}} \;=\; \mathbf{F}\,\hat{\alpha} \in \mathbb{C}^M$, and use the normalized estimate as a Maximum Ratio Transmission (MRT) precoder $\hat{\mathbf{v}} \;=\; \frac{\hat{\mathbf{h}}}{\|\hat{\mathbf{h}}\|_2}$. The data-phase receive model is thus

$$y \;=\; \mathbf{h}^H \hat{\mathbf{v}}\, x + w. \tag{3}$$

2.3 Data Queuing Model

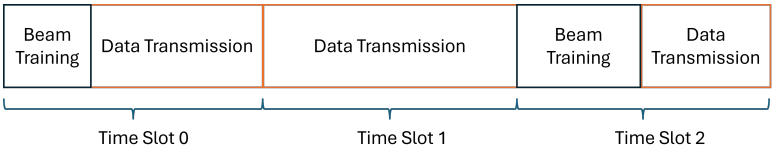

Figure 2: Demonstration of behavior in each time slot.

In this section, we specify the queuing dynamics in each time slot. Suppose the slot duration is T_s. Owing to user mobility, the channel varies across consecutive slots. In each time slot, beam training is performed first (if needed) and then data is transmitted based on the estimated channel. The time that beam training consumes is proportional to the number of pilot measurements in our frame structure, which is illustrated in Fig. 2. Denote by m_t the number of pilot measurements used in slot t and by τ_{ov} the per-pilot overhead (seconds per pilot). The training duration in slot t is therefore

$$T_{\text{train},t} \;=\; m_t\, \tau_{\text{ov}}. \tag{4}$$

The remaining time in the slot is available for data transmission:

$$T_{\text{data},t} \;=\; T_s - T_{\text{train},t}. \tag{5}$$

According to Eq. (3), let $\text{SNR}_t = p_t |\mathbf{h}^H \mathbf{v}|^2/\sigma^2$ denote the instantaneous received signal-to-noise ratio (SNR), where p_t is the power for data transmission at the gNB. Then, $R_t = \log_2(1 + \text{SNR}_t)$ denotes the achievable rate in the data phase. With system bandwidth W (Hz) the number of bits that can be delivered in the data phase of slot t is

$$\psi_t \;=\; T_{\text{data},t}\, W R_t. \tag{6}$$

Let q_t be the queue length (bits) at the beginning of slot t. Let ℓ_t be the new arrivals (bits) that arrive within slot t, which follows a Poisson Distribution. The buffer has finite capacity $Q_{\max}$; any excess arrivals that would push the buffer beyond $Q_{\max}$ are dropped. The queue update is thus written as:

$$q_{t+1} \;=\; \min\Big\{Q_{\max},\; \max\{q_t - \psi_t, 0\} + \ell_t\Big\}, \tag{7}$$

and the overflows (dropped bits) in slot t can be expressed as

$$d_t \;=\; \max\big\{\, q_t - \psi_t + \ell_t - Q_{\max} \,\big\}. \tag{8}$$

3 Problem Formulation

In this section, we introduce the proposed cross-layer decision model.

We model the decision problem as a partially observable Markov decision process (POMDP) (31) and solve it with Proximal Policy Optimization (PPO) (32) as given in Section D. To jointly decide the beam training and data transmission procedure, the model needs to output the following actions $a_t = \big(b_t,\ m_t,\ K_t,\ p_t\big),$ where $b_t \in \{0, 1\}$ indicates whether CS training is performed in slot t, m_t is the pilot budget (used only if $b_t = 1$), K_t is the TSP target sparsity, and p_t is the data-phase transmit power.

Since we jointly consider the PHY and MAC design, the agent cannot observe the true instantaneous channel state before making a decision. This is in contrast to prior work in which the channel state is assumed to be known (33; 34). In this paper, the agent observes a compact tuple that summarizes queueing and recent history. Denote the agent's observation at the start of slot t by $s_t = \big(q_t,\ \tau_t,\ I_t\big)$, where q_t is the queue length and τ_t is the age (slots since last training). To avoid a degenerate policy that never learns, we enforce $b_t = 1$ at $t = 0$ or whenever $\tau_t > \tau_{\text{train}}$. A reward-design alternative could eliminate this heuristic, but we leave that as future work. I_t is a short history window of recent T_{age} measurement and training tuples:

$$I_t = \big((b_{t-T_{\text{age}}}, m_{t-T_{\text{age}}}, K_{t-T_{\text{age}}}, R'_{t-T_{\text{age}}}), \cdots, (b_{t-1}, m_{t-1}, K_{t-1}, R'_{t-1})\big), \tag{9}$$

where $R' = \log_2(1 + |\mathbf{h}^H\mathbf{v}|^2/\sigma^2)$ is the original rate without influence of the transmission power. We obtain transition samples by observing the observation s_t, executing the action a_t chosen by PPO in the simulator (or system) and observing the resulting next observation s_{t+1} and reward r_t. Concretely, starting from s_t and applying a_t we first compute the training and data transmission durations via Eq. (4) and Eq. (5), respectively. If $b_t = 1$ the environment returns measurements $\mathbf{y}_t$ and the TSP recovery $\hat{\alpha}_t$, from which the channel estimate $\hat{\mathbf{h}}_t$ and the beamformer are constructed. The data-phase SNR and delivered service are then computed via Eq. (6), and the next queue state is determined via Eq. (7).

We design the immediate reward to reflect energy, delay, and overflow costs. The beam training process uses maximum power (normalized to 1) for beam sweeping. Define the per-slot energy as

$$E_t \ = \ E_{\text{train}}(m_t) \ + \ p_t\, T_{\text{data},t} \ \text{ and } \ E_{\text{train}}(m_t) = m_t \tau_{\text{ov}}, \tag{10}$$

where m_t is the pilot budget, τ_{ov} is the time consumption of one single beam for beam sweeping, $p_t \in [0, 1]$ is the normalized transmit power, and $T_{\text{data},t}$ is given by Eq. (5). The per-slot reward is then

$$r_t \ = \ -(E_t \ + \ \lambda_Q\, q_{t+1} \ + \ \lambda_{\text{drop}}\, d_t), \tag{11}$$

with $\lambda_Q > 0$ weighting the delay penalty via the next-queue length q_{t+1} from Eq. (7), and $\lambda_{\text{drop}} \geq 0$ weighting the overflow penalty through the dropped bits d_t in Eq. (8). Thus, Eq. (11) is equivalently a cost $c_t = E_t + \lambda_Q q_{t+1} + \lambda_{\text{drop}} d_t$ with $r_t = -c_t$, where the energy term captures training and data-phase expenditure and the queueing terms capture latency and reliability consequences of the current decision (b_t, m_t, K_t, p_t).

4 Experiments

In this session, we provide the simulation results and discussion. To evaluate the proposed model, we built a custom Gymnasium (35) environment based on the system model presented in Section 2. Our learning agent is based on StableBaseline3 PPO (36). We discretize continuous components (e.g., p_t) and use a shared MLP trunk with separate policy/value heads. The central frequency is $f_c = 30$ GHz and antenna size is $M_x = 128, M_z = 8$. The detailed parameter selection, component discretization, and experiment setting can be found in Section B. Let an episode contain T_{tol} slots. We report the following metrics, and first form episode-level quantities, then report across-episode mean $\pm$ standard deviation.

Achievable rate. The per-episode mean rate is $\tilde{R} = (1/T_{\text{tol}}) \sum_{t=1}^{T_{\text{tol}}} \psi_t/T_s$ (bps).

Beamforming-gain ratio. Per-slot $\rho_t = |\mathbf{v}_t^H \mathbf{h}_t|^2/\|\mathbf{h}_t\|^2 \in [0, 1]$, with time average $\rho = (1/T_{\text{tol}}) \sum_{t=1}^{T_{\text{tol}}} \rho_t$.

Overflow rate. With the queue update in Eq. (7), and per-slot overflow d_t defined in Eq. (8), we report the bits-based overflow fraction over T_{tol} slots as $\mathrm{Ov}(\%) = 100 \times \left(\sum_{t=1}^{T_{\mathrm{tol}}} d_t\right) / \left(\sum_{t=1}^{T_{\mathrm{tol}}} \ell_t\right)$, where ℓ_t is the number of arrived bits in slot t.

Energy consumption. The mean energy is $E = (1/T_{\mathrm{tol}}) \sum_{t=1}^{T_{\mathrm{tol}}} E_t$.

Train time fraction. Using the per-slot training duration $T_{\mathrm{train},t}$ from Eq. (4) and the slot length T_s from Eq. (5), we report $\mathrm{TTF}(\%) \;=\; 100 \times (1/T_{\mathrm{tol}}) \sum_{t=1}^{T_{\mathrm{tol}}} \left(T_{\mathrm{train},t}/T_s\right)$,

4.1 Comparison with Baselines

We compare our method against three baselines.

5G NR (DFT codebook). Following 5G NR beam management, we perform beam sweeping over a DFT codebook and select the beam with the highest instantaneous SNR for data transmission. Beam training is executed periodically every τ_{train} slots. The data transmission power is fixed at the same maximum power $P_{\max}$ used by our method.

Near-field–improved 5G NR. This baseline uses the same training schedule and power setting as above but replaces the DFT codebook with the near-field codebook from (37), which extends (13) to UPA.

Full CSI (oracle upper bound). We assume perfect channel knowledge and use the matched filter/maximum-ratio beam $\mathbf{v}_t^\star = \mathbf{h}_t / \|\mathbf{h}_t\|$. No beam training is needed (zero time/energy overhead) and data power is set to $P_{\max}$. This serves as an upper bound on performance.

Table 1: Baseline comparison.

Method	$\bar{\tilde{R}}$ (Mbps)	$\bar{\rho}$	$\overline{\mathrm{Ov}}$ (%)	$\bar{E}$	$\overline{\mathrm{TTF}}$ (%)
DFT	26.9 ± 4.1	0.317 ± 0.104	6.7 ± 4.7	0.001 ± 0.000	5.0 ± 0.0
Near-field	31.0 ± 6.3	0.410 ± 0.101	9.3 ± 8.9	0.001 ± 0.000	9.3 ± 0.0
Full CSI	57.1 ± 7.3	1.000 ± 0.000	0.4 ± 2.2	0.001 ± 0.000	0.0 ± 0.0
Proposed	49.9 ± 34.0	0.853 ± 0.052	1.5 ± 6.1	0.001 ± 0.000	3.2 ± 0.0

Relative to the two 5G-NR style baselines, the proposed method increases throughput to 49.9 Mbps, which corresponds to a $+85.5\%$ improvement over DFT sweeping at 26.9 Mbps and $+61.0\%$ over the near-field codebook at 31.0 Mbps. The beamforming-gain ratio is $\rho = 0.853$, which is $2.69\times$ the DFT value 0.317 and $2.08\times$ the near-field value 0.410. Queueing performance improves accordingly: the overflow rate is reduced to 1.5%, i.e., a decrease of 5.2 percentage points relative to DFT (6.7%) and 7.8 percentage points relative to near-field (9.3%). Training overhead is also lower at 3.2%, compared with 5.0% for DFT and 9.3% for near-field. A gap to the full-CSI upper bound remains: the proposed rate achieves $49.9/57.1 \approx 87.4\%$ of the oracle throughput. Energy consumption is identical across methods in this setup because all policies transmit at maximum power (as in the 5G-NR baselines). Exploring the energy–delay trade-off by adjusting the cost weights in Eq. (11) is left for future work.

4.2 Ablation Experiments

We conduct two ablations to quantify the contribution of temporal history in the observation and periodic beam training. In **No history**, we remove the history by setting $T_{\mathrm{age}} = 0$. In **No train**, we eliminate periodic training and train only at $t = 0$ (i.e., no τ_{train}). The full model (**w/o ablation**) uses both components.

Table 2: Ablation comparison.

Method	$\bar{\tilde{R}}$ (Mbps)	$\bar{\rho}$	$\overline{\mathrm{Ov}}$ (%)	$\bar{E}$	$\overline{\mathrm{TTF}}$ (%)
No history	36.8 ± 5.3	0.566 ± 0.065	5.3 ± 5.4	0.001 ± 0.000	3.6 ± 0.0
No train	35.2 ± 23.8	0.374 ± 0.083	6.4 ± 12.4	0.001 ± 0.000	0.0 ± 0.0
w/o ablation	49.9 ± 34.0	0.853 ± 0.052	1.5 ± 6.1	0.001 ± 0.000	3.2 ± 0.0

Energy consumption is identical across methods in this setup. The full model (w/o ablation) achieves 49.9 Mbps with $\rho = 0.853$ and an overflow rate of 1.5% at 3.2% training time. Removing history

reduces the rate to 36.8 Mbps, a 26.3% drop relative to the full model, and lowers beamforming-gain ratio to $\rho = 0.566$ (a 33.6% decrease). The overflow rate increases by 3.8 percentage points to 5.3%; training time is slightly higher at 3.6%. Eliminating periodic training is more detrimental: the rate falls to 35.2 Mbps (29.5% below full), beamforming-gain ratio drops to $\rho = 0.374$ (56.1% decrease), and the overflow rate rises by 4.9 percentage points to 6.4% while using 0% training time. These results indicate that both temporal context and periodic training contribute to performance, with training providing the larger share of the gain in both throughput and alignment.

5 Conclusion

We presented a cross-layer design for near-field mmWave tailored to delay-sensitive traffic under explicit energy constraints. By jointly optimizing beam alignment, power, and queue-aware scheduling, our method balances training overhead and data transmission time, yielding higher achievable rate and lower overflow than DFT and near-field codebook baselines while approaching full-CSI performance. Future work includes extending to multi-user settings, formulating simpler greedy methods to reduce the model training and evaluation time, exploiting ML optimization algorithms with higher sample efficiency and lower training convergence time (18; 38), and integration of application-centric objectives for further impact on next generation wireless IoT and XR application systems.

Acknowledgments

This material is based upon work supported in part by the National Science Foundation (NSF) under Grant Nos. CNS-2106150, CNS-2032033, CNS-2346528, and ECCS-2512911.

References

[1] Saad, W., M. Bennis, M. Chen. A vision of 6G wireless systems: Applications, trends, technologies, and open research problems. *IEEE Netw.*, 34(3):134–142, May 2020.

[2] Zhang, Z., Y. Xiao, Z. Ma, et al. 6G wireless networks: Vision, requirements, architecture, and key technologies. *IEEE Veh. Technol. Mag.*, 14(3):28–41, Jul. 2019.

[3] Akyildiz, I. F., J. M. Jornet, C. Han. Terahertz band: Next frontier for wireless communications. *Phys. Commun.*, 12:16–32, Sep. 2014.

[4] Björnson, E., L. Sanguinetti, H. Wymeersch, et al. Massive MIMO is a reality—what is next?: Five promising research directions for antenna arrays. *Digit. Signal Process.*, 94:3–20, Nov. 2019.

[5] Lu, H., Y. Zeng, C. You, et al. A tutorial on near-field xl-MIMO communications toward 6G. *IEEE Commun. Surveys Tuts.*, 26(4):2213–2257, Apr. 2024.

[6] Liu, Y., Z. Wang, J. Xu, et al. Near-field communications: A tutorial review. *IEEE Open J. Commun. Soc.*, 4:1999–2049, Aug. 2023.

[7] Liu, Y., C.-X. Wang, H. Chang, et al. A novel non-stationary 6G UAV channel model for maritime communications. *IEEE J. Sel. Areas Commun.*, 39(10):2992–3005, Oct. 2021.

[8] Baduge, G. A., M. Vaezi, J. K. Dassanayake, et al. Frequency range 3 for ISAC in 6G: Potentials and challenges, 2025.

[9] Lu, H., Y. Zeng. How does performance scale with antenna number for extremely large-scale MIMO? In *Proc. IEEE ICC*, pages 1–6. Jun. 2021.

[10] —. Communicating with extremely large-scale array/surface: Unified modeling and performance analysis. *IEEE Trans. Wireless Commun.*, 21(6):4039–4053, Jun. 2022.

[11] Wang, Z., R. Kiran, S. Tsai, et al. Low-complexity near-field beam training with dft codebook based on beam pattern analysis. *arXiv preprint arXiv:2503.21954*, Mar. 2025.

[12] Wei, X., L. Dai. Channel estimation for extremely large-scale massive MIMO: Far-field, near-field, or hybrid-field? *IEEE Commun. Lett.*, 26(1):177–181, Jan. 2022.

[13] Cui, M., L. Dai. Channel estimation for extremely large-scale MIMO: Far-field or near-field? *IEEE Trans. Commun.*, 70(4):2663–2677, Apr. 2022.

[14] Ziao, Q., Y. Haifan. A review of codebooks for csi feedback in 5G new radio and beyond. *China Communications*, 22(2):112–127, 2025.

[15] Wang, Z., R. Kiran, J. Nair, et al. Sparsity-aware near-field beam training via multi-beam combination. *arXiv preprint arXiv:2505.08267*, 2025.

[16] Lei, W., D. Zhang, Y. Ye, et al. Joint beam training and data transmission control for mmWave delay-sensitive communications: A parallel reinforcement learning approach. *IEEE Journal of Selected Topics in Signal Processing*, 16(3):447–459, 2022.

[17] Mastronarde, N., N. Sharma, J. Chakareski. Improving data-driven reinforcement learning in wireless IoT systems using domain knowledge. *IEEE Communications Magazine*, 59(11):95–101, 2021.

[18] Sharma, N., N. Mastronarde, J. Chakareski. Accelerated structure-aware reinforcement learning for delay-sensitive energy harvesting wireless sensors. *IEEE Trans. Signal Processing*, 68(1):1409–1424, 2020.

[19] Shokri-Ghadikolaei, H., L. Gkatzikis, C. Fischione. Beam-searching and transmission scheduling in millimeter wave communications. In *2015 IEEE International Conference on Communications (ICC)*, pages 1292–1297. IEEE, London, UK, 2015.

[20] Huang, A., K.-H. Lin, H.-Y. Wei. Beam-aware cross-layer DRX design for 5G millimeter wave communication system. *IEEE Access*, 8:77604–77617, 2020.

[21] Pan, M.-S., Y.-X. Chen. Beam-aware scheduling for 5G millimeter wave networks with discontinuous reception and dual connectivity user equipments. *IEEE Systems Journal*, pages 1–12, 2023. Early access.

[22] Li, Y., L. Su, T. Wei, et al. Location-aware dynamic beam scheduling for maritime communication systems. In *2018 10th International Conference on Communications, Circuits and Systems (ICCCAS)*, pages 265–268. IEEE, Chengdu, China, 2018.

[23] Gupta, S., J. Chakareski, P. Popovski. mmWave networking and edge computing for scalable 360-degree video multi-user virtual reality. *IEEE Trans. Image Processing*, 32:377–391, 2023.

[24] Badnava, B., J. Chakareski, M. Hashemi. Neural-enhanced rate adaptation and computation distribution for emerging mmWave multi-user 3D video streaming systems. *IEEE Trans. Multimedia*, 27:7125–7136, 2025.

[25] Chakareski, J., M. Khan, T. Ropitault, et al. Millimeter wave and free-space-optics for future dual-connectivity 6DOF mobile multi-user VR streaming. *ACM Trans. Multimedia Computing Communications and Applications*, 19(2):57:1–25, 2023.

[26] Chakareski, J., M. Khan. Live 360° video streaming to heterogeneous clients in 5G networks. *IEEE Trans. Multimedia*, 26:8860–8873, 2024.

[27] Srinivasan, S., S. Shippey, E. Aryafar, et al. FBDT: Sum-throughput achieving forward and backward data transmission across multi-RAT networks. *IEEE Trans. Mobile Computing*, 24(10):10069–10084, 2025.

[28] Selvan, K. T., R. Janaswamy. Fraunhofer and fresnel distances: Unified derivation for aperture antennas. *IEEE Antennas Propag. Mag.*, 59(4):12–15, Dec. 2017.

[29] Dai, W., O. Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *IEEE Transactions on Information Theory*, 55(5):2230–2249, 2009.

[30] Candes, E., T. Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

[31] Åström, K. J. Optimal control of markov processes with incomplete state information I. *Journal of Mathematical Analysis and Applications*, 10:174–205, 1965.

[32] Schulman, J., F. Wolski, P. Dhariwal, et al. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[33] Sharma, N., S. Zhang, S. R. Somayajula Venkata, et al. Deep reinforcement learning for delay-sensitive lte downlink scheduling. In *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1–6. 2020.

[34] Omer, A., F. Malandra, J. Chakareski, et al. Performance evaluation of 5G delay-sensitive single-carrier multi-user downlink scheduling. In *2023 IEEE 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1–6. 2023.

[35] Towers, M., A. Kwiatkowski, J. Terry, et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.

[36] Raffin, A., A. Hill, A. Gleave, et al. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

[37] Wu, Z., L. Dai. Multiple Access for Near-Field Communications: SDMA or LDMA? *IEEE Journal on Selected Areas in Communications*, 41(6):1918–1935, 2023.

[38] Felizardo, L. K., E. Fadda, P. Brandimarte, et al. A reinforcement learning method for environments with stochastic variables: Post-decision proximal policy optimization with dual critic networks, 2025.

A DFT codebook implementation

In our implementation, we set the DFT grid sizes equal to the UPA dimensions, i.e., $P = M_x$ and $Q = M_z$. Let $M = M_x M_z$ and stack the UPA elements into the column vector $\mathbf{h} \in \mathbb{C}^M$ using column-major order (index m varies fastest inside each n). Recall δ_n and δ_m as defined above. Define the 1-D spatial frequency grids

$$u_p = \frac{2p}{M_x} - 1, \quad p = 0, \ldots, M_x - 1, \qquad w_q = \frac{2q}{M_z} - 1, \quad q = 0, \ldots, M_z - 1.$$

For each grid pair (u_p, w_q) the corresponding normalized steering column $\mathbf{f}_{p,q} \in \mathbb{C}^M$ has entries

$$\mathbf{f}_{p,q}[m, n] = \frac{1}{\sqrt{M}} \exp\big(- j\pi(\delta_n u_p + \delta_m w_q)\big), \qquad m = 1, \ldots, M_z,\ n = 1, \ldots, M_x,$$

where $\mathbf{f}_{p,q}[m, n]$ denotes the entry at element (m, n) in the same stacking order used for $\mathbf{h}$. The 2-D DFT dictionary (codebook) is the $M \times M$ matrix formed by concatenating these columns,

$$\mathbf{F} = \big[\, \mathbf{f}_{0,0}\ \mathbf{f}_{1,0}\ \ldots\ \mathbf{f}_{M_x-1,M_z-1} \,\big] \in \mathbb{C}^{M \times M}.$$

B Main system parameters

We evaluate all methods(Section 2), each run for E=40 episodes of T=5000 slots under distinct random seeds ($\texttt{seed}_0$ + episode index). Unless otherwise stated, policies are evaluated in deterministic (greedy) mode (`deterministic=True` in PPO), and all PHY/MAC/system parameters are kept identical across methods. Baselines use the settings in Section 4.1 (DFT or near-field codebook sweeping every τ_{train} slots with p_t=$P_{\max}$; full-CSI uses matched filtering with no training overhead).

C Compressive sensing based beam training

For algorithmic recovery we equivalently work with the column-form by taking Hermitian transpose:

$$\tilde{\mathbf{y}} \;=\; \mathbf{y}^H \in \mathbb{C}^{m \times 1}, \qquad \tilde{\mathbf{A}} \;=\; \mathbf{A}^H \in \mathbb{C}^{m \times M}, \qquad \tilde{\mathbf{w}} \;=\; \mathbf{w}^H \in \mathbb{C}^{m \times 1}$$

which yields the standard CS model

$$\tilde{\mathbf{y}} \;=\; \tilde{\mathbf{A}}\, \alpha + \tilde{\mathbf{w}}.$$

The detailed algorithm is in Algorithm 1.

Table 3: Main system parameters.

Parameter	Meaning	Value
f_c	Carrier (central) frequency	$30\,\mathrm{GHz}$
λ	Wavelength ($\lambda = c/f_c$)	$10\,\mathrm{mm}$
d	Antenna spacing	$\lambda/2 = 5\,\mathrm{mm}$
$M_x \times M_z$	UPA size (horizontal $\times$ vertical)	128×8 ($M = 1024$ elements)
W	System bandwidth	$20\,\mathrm{MHz}$
T_s	Slot duration	$1\,\mathrm{ms}$
τ_{train}	Beam-training period	10 slots (default)
τ_{ov}	Time for one overhead in beam training	$1/2048 \times 10^{-4}\,\mathrm{ms}$
L_{path}	Number of channel paths	3
σ^2	Noise variance	5.2×10^{-10}
T_{age}	History window length in observation	64 slots
$Q_{\max}$	Queue capacity	80,000 bits
λ_{arr}	Packet arrival intensity	2000 pkts/s
b_{pkt}	Bits per packet	6000 bits
$\mathbb{E}[\ell_t]$	Mean arrival per slot ($\lambda_{\mathrm{arr}} b_{\mathrm{pkt}} T_s$)	12,000 bits/slot
T	Maximum iterations for TSP	1
P_{level}	Power discretization levels	10 (i.e., $p \in \{0.1, \ldots, 1.0\}$, $P_{\max} = 1$)
m_{level}	Pilot/overhead budget levels	10 (ratios; $m_i = \lceil \frac{i+1}{m_{\mathrm{level}}} M \rceil, i = 0, \ldots, 9$, with $M = M_x M_z$)
K_{level}	Sparsity levels	10 (ratios; $K_i = \max\{1, \lceil \frac{i+1}{2K_{\mathrm{level}}} m_t \rceil\}, i = 0, \ldots, 9$)

Algorithm 1 TSP

Require: $\mathbf{A} \in \mathbb{C}^{M \times m}$, row-measurements $\mathbf{y} \in \mathbb{C}^{1 \times m}$, target sparsity k, max iterations T

$\tilde{\mathbf{y}} \leftarrow \mathbf{y}^H, \quad \tilde{\mathbf{A}} \leftarrow \mathbf{A}^H$
$\mathbf{z} \leftarrow \tilde{\mathbf{A}}^H \tilde{\mathbf{y}}$
$\widehat{\mathcal{S}} \leftarrow \mathrm{TopK}(|\mathbf{z}|, k)$ (TopK: indices of the k largest $|\mathbf{z}_i|$)
for $t = 1$ **to** T **do**
  $\hat{\alpha}_{\widehat{\mathcal{S}}} \leftarrow \tilde{\mathbf{A}}_{\widehat{\mathcal{S}}}^{\dagger} \tilde{\mathbf{y}}$ (LS on current support (pseudoinverse))
  $\mathbf{r} \leftarrow \tilde{\mathbf{y}} - \tilde{\mathbf{A}}_{\widehat{\mathcal{S}}} \hat{\alpha}_{\widehat{\mathcal{S}}}$
  $\mathbf{z} \leftarrow \tilde{\mathbf{A}}^H \mathbf{r}$
  $\widehat{\mathcal{S}}_{\mathrm{new}} \leftarrow \widehat{\mathcal{S}} \cup \mathrm{TopK}(|\mathbf{z}|, k)$
  if $\widehat{\mathcal{S}}_{\mathrm{new}} = \widehat{\mathcal{S}}$ **then**
    break (Converged: support unchanged)
  end if
  $\widehat{\mathcal{S}} \leftarrow \widehat{\mathcal{S}}_{\mathrm{new}}$
end for
$\hat{\alpha} \leftarrow \mathbf{0} \in \mathbb{C}^M; \quad \hat{\alpha}_{\widehat{\mathcal{S}}} \leftarrow \tilde{\mathbf{A}}_{\widehat{\mathcal{S}}}^{\dagger} \tilde{\mathbf{y}}$
return $\hat{\alpha}$, $\widehat{\mathcal{S}}$

D PPO

We denote policy and value parameters by $\boldsymbol{\omega}$ and $\boldsymbol{\nu}$, respectively.

We estimate advantages with generalized advantage estimation (GAE) using a critic $V_{\boldsymbol{\nu}}$:

$$\delta_t \;=\; r_t + \gamma V_{\boldsymbol{\nu}}(s_{t+1}) - V_{\boldsymbol{\nu}}(s_t), \qquad \hat{A}_t \;=\; \sum_{\ell=0}^{L-1} (\gamma\, \lambda_{\mathrm{GAE}})^{\ell}\, \delta_{t+\ell}. \tag{12}$$

The multi-discrete action $a_t = (b_t, m_t, K_t, p_t)$ is modeled by a factorized categorical policy

$$\pi_{\boldsymbol{\omega}}(a_t \mid s_t) \;=\; \prod_{j=1}^{4} \pi_{\boldsymbol{\omega}}^{(j)}\big(a_t^{(j)} \mid s_t\big). \tag{13}$$

PPO maximizes the clipped surrogate with ratio $\rho_t(\boldsymbol{\omega}) = \frac{\pi_{\boldsymbol{\omega}}(a_t|s_t)}{\pi_{\boldsymbol{\omega}_{\text{old}}}(a_t|s_t)}$:

$$\mathcal{L}_t^{\text{CLIP}}(\boldsymbol{\omega}) = \min\Big(\rho_t(\boldsymbol{\omega})\,\hat{A}_t,\ \text{clip}\big(\rho_t(\boldsymbol{\omega}), 1-\epsilon, 1+\epsilon\big)\,\hat{A}_t\Big), \tag{14}$$

and we train an actor–critic with

$$L(\boldsymbol{\omega}, \boldsymbol{\nu}) \ = \ -\mathbb{E}_t[\mathcal{L}_t^{\text{CLIP}}] \ + \ c_1\,\mathbb{E}_t\big[\big(V_{\boldsymbol{\nu}}(s_t) - r'_t\big)^2\big] \ - \ c_2\,\mathbb{E}_t\big[\mathcal{H}\big(\pi_{\boldsymbol{\omega}}(\cdot \mid s_t)\big)\big], \tag{15}$$

where $c_1, c_2 > 0$ weight value regression and entropy regularization. The target r'_t is the truncated, bootstrapped return Rollouts are used to compute $\hat{A}_t$ in Eq. (12) and to optimize Eq. (15) via minibatch SGD.

Training schedule (total rounds). Unless otherwise noted, we use $\gamma = 0.99$ and $\lambda_{\text{GAE}} = 0.95$. Training proceeds for N_{upd} PPO updates, each collecting n_{steps} transitions per environment over N_{env} parallel environments. The total number of environment steps is

$$T_{\text{tot}} \ = \ N_{\text{upd}}\, n_{\text{steps}}\, N_{\text{env}}.$$

In our reported runs we used $T_{\text{tot}} \ = \ 15.16$M environment steps. Each update performs K_{epoch} epochs of minibatch SGD with minibatch size M_{mb}. The remaining hyperparameters are standard PPO: clipping parameter ϵ, learning rate η with Adam, value loss weight c_1, and entropy weight c_2.

Network architecture. Observations are fed to a shared multilayer perceptron (MLP) with two hidden layers of widths $(128, 64)$ and elementwise nonlinearity (ReLU or Tanh). From the shared trunk, we branch into: (i) a policy head that outputs concatenated logits for the four categorical factors

$$\underbrace{2}_{b_t} \ + \ \underbrace{m_{\text{level}}}_{m_t} \ + \ \underbrace{K_{\text{level}}}_{K_t} \ + \ \underbrace{P_{\text{level}}}_{p_t} \ = \ 32,$$

which are then partitioned to form $\pi_{\boldsymbol{\omega}}^{(j)}$ in Eq. (13); and (ii) a value head that outputs the scalar $V_{\boldsymbol{\nu}}(s_t)$. The input size (observation size) for the full model is $4T_{\text{age}} + 2$ (the 4 features per slot from the history window plus q_t and τ_t); with $T_{\text{age}} = 64$ this equals 258. For the no-history ablation we set $T_{\text{age}} = 0$, giving an input size of $4 \cdot 0 + 2 = 2$. Both actor and critic use the shared $(128, 64)$ trunk with separate linear output layers, advantage normalization, and gradient clipping.