Getting Ready for Deployment: Transformer-Enabled Robust Adaptive Traffic Signal Control

Xiaoyu Wang*,1, Ilia Smirnov*,1, Scott Sanner[2], and Baher Abdulhai[1]

***Abstract*— Urban traffic signal control must contend with rapidly changing demand patterns, limited sensor coverage, and strict safety regulations — challenges often overlooked by lab-focused reinforcement learning (RL) research. We introduce eMARLIN-Transformer-Robust, a decentralized multi-agent RL framework built for real-world deployment. We frame these issues as partial observability and out-of-distribution challenges. To reduce partial observability, each agent employs a Transformer encoder over a brief history of local observations augmented by neighbor embeddings. The architecture captures temporal context and extends agent's effective field of view. To achieve out-of-distribution robustness, we apply domain randomization techniques, exposing agents to diverse traffic scenarios to prevent overfitting.**

In a simulated field test spanning 75 out-of-training scenarios (225 runs), eMARLIN-Transformer-Robust reduces total stop delay by $17.7\% \sim 27.8\%$ compared to the expert-tuned industry-standard TransSuite system. An ablation study shows that robust training further improves performance by 12.2% over agents lacking it. These results demonstrate that only the combination of Transformer-based history encoding and robust training yields a single, adaptive policy capable of seamless, plan-free operation across diverse, real-world conditions.

I. INTRODUCTION

Urban traffic congestion remains one of the most pressing challenges for modern cities [1], [2], driving the need for advanced signal control systems that can adapt in real time to rapidly changing conditions [3], [4]. While reinforcement learning (RL) has delivered dramatic delay reductions in laboratory studies, most algorithmic advances have been validated under idealized settings that overlook critical real-world complexities.

In practice, deploying RL for Adaptive Traffic Signal Control (ATSC) faces two deeply intertwined hurdles. First, real networks exhibit highly dynamic, spatiotemporal variations in demand. RL agents trained on a limited set of profiles often encounter new patterns in deployment, leading to overfitting and degraded performance — an instance of the Out-Of-Distribution (OOD) problem. Ensuring OOD robustness is therefore our primary objective. Second, real-world controllers suffer from partial observability (PO): limited sensor coverage prevents agents from accessing the full system state, further exacerbating OOD issues by interfering their ability to infer true traffic dynamics.

Complicating matters, practical deployments must also adhere to hard operational constraints to guarantee safety and equity. These policy constraints narrow the action space in ways that laboratory benchmarks rarely capture (e.g., many studies allow arbitrary phase transitions [5]; see Section III-B for details). Agents must thus learn not only to infer unobserved traffic contexts but also to navigate a restricted decision set, reacting proactively and adapting seamlessly to real-world demands.

Some research isolates individual intersections and views them families of models parameterized by key traffic metrics (e.g., flow rate, speed limit) [6]. With this concept, [7] tries to train a unified controller that explicitly conditions on a key metric. By modeling the performance gap with respect to the key metric, it guides efficient task selection in training. While elegant, this approach still relies on a single contextual variable and doesn't naturally capture the full richness of interactions between neighboring intersections under wildly varying conditions.

In this study, we adopt a different perspective. We formulate ATSC as a decentralized Partially Observable Markov Decision Process (Dec-POMDP), optionally integrating hard policy constraints for safety and regulatory compliance. To tackle both OOD and PO within this framework, we introduce eMARLIN-Transformer-Robust, a decentralized multi-agent system that combines Transformer-based history encoding with a comprehensive robust training pipeline. The eMARLIN-Transformer equips each agent with a self-attention encoder that processes a short history of local observations and compact embeddings from neighboring intersections, mitigating PO by capturing temporal context and extending effective observability. The Robust Training pipeline built on domain randomization techniques exposes the agents to a wide spectrum of simulated traffic conditions. This strategy transforms extrapolation into interpolation, ensuring that the final policy generalizes reliably across unforeseen scenarios.

In an extensive simulated field test covering 75 out-of-training scenarios (225 runs), eMARLIN-Transformer-Robust consistently outperforms both time-of-day specialist agents and the industry-standard TransSuite baseline. These results confirm that both Transformer encoding and robust training are necessary and together sufficient to produce a single, unified policy capable of seamless real-time adaptation without plan switching.

* Equal contribution

[1]Xiaoyu Wang, Ilia Smirnov, and Baher Abdulhai are with the Department of Civil & Mineral Engineering, University of Toronto, Canada cnxiaoyu.wang@mail.utoronto.ca, {ilia.smirnov, baher.abdulhai}@utoronto.ca

[2]Scott Sanner is with the Department of Mechanical & Industrial Engineering, University of Toronto, Canada ssanner@mie.utoronto.ca

The authors thank Aimsun for providing research licenses for the Aimsun microsimulator and gratefully acknowledge the City of Toronto for their collaboration on this pilot project.

II. PRELIMINARIES

In this section, we briefly review the foundations of reinforcement learning and the Transformer architecture.

A. Reinforcement Learning

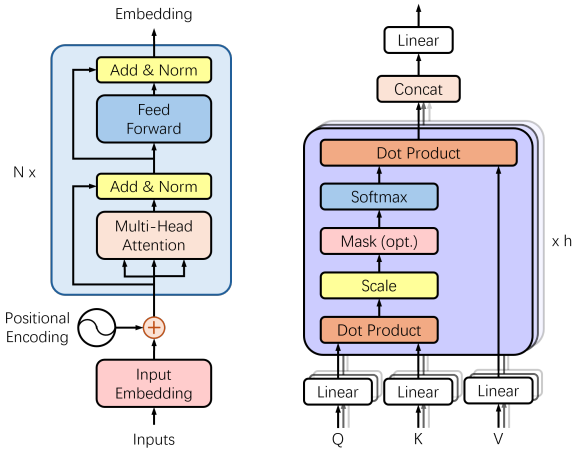
RL tackles problems modeled as Markov Decision Processes (MDPs) comprising state, action, transition probability, immediate reward, and discount factor components, denoted as $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$. A return function calculates the total discounted reward over time, defined as $G_t = \sum_k \gamma^k r_{t+k}$. Agents typically operate with Markov policies $\pi(a|s)$, making decisions based on current states. The object of RL is to learn a policy π that maximizes the return G_t without knowing the true dynamics $\mathcal{T}$ of the model.

Q-learning estimates the optimal action-value function $Q^*(s,a)$ via the Bellman optimality update:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right]. \tag{1}$$

Deep Q-Networks (DQNs) approximate the look-up Q-table with neural networks (NNs) [8], enabling generalization to unseen or similar states, granting DQNs the capability of handling problems with large search spaces, such as traffic management problems.

B. Transformers



(a) The architecture of a Transformer encoder model. Adapted from [9].

(b) The multi-head attention layer, where the purple box defines the scaled dot-product attention.

Fig. 1: The Transformer architecture and its breakdowns.

In machine learning, attention mechanisms dynamically compute weights to focus on relevant parts of input sequences or spatial dimensions, providing powerful feature abstraction capabilities. The Transformer architecture [9] employs a self-attention mechanism to model dependencies across an input sequence without recurrence, being efficient in capturing long-range correlations. Core to the Transformer's innovation is its scaled dot-product attention and flexible input design. This work particularly examines the Transformer encoder, as illustrated in Figure 1a, highlighting two key innovations.

a) Input encoding: Each raw observation is first projected into a d_{model}-dimensional embedding to reduce dimensionality and facilitate representation learning. Transformers handle sequential information by incorporating positional encoding, typically achieved through sinusoidal mapping. We then combine it with the observation embedding to retain sequence order information, producing the final input to the encoder layers (Figure 1a).

b) The Transformer block: The key architectural motif of the Transformer is the scaled dot-product (or QKV) attention mechanism, illustrated in Figure 1b. Three matrices $Q, K \in \mathbb{R}^{d_i \times d_k}$, and $V \in \mathbb{R}^{d_i \times d_v}$ are linearly transformed from the input $X \in \mathbb{R}^{d_i \times d_{model}}$, projecting them into distinct subspaces, allowing the attention mechanism to selectively weigh and aggregate information based on learned token relationships. The output is defined as:

$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^\top}{\sqrt{d_k}})V, \tag{2}$$

which is a combination of V that is weighted by the intensity of attention between any pair of two slices in the input sequence. Transformers further stack and simultaneously train multiple instances of the dot-product attention module, attending to different aspects of the input, enabling a more nuanced understanding of the data. The multi-head mechanism also provides scalability in handling large-scale data sets and complex tasks.

III. PROBLEM FORMULATION

A. Partially Observable MDP

Standard MDPs assume full state observability, which is impractical for real-world traffic control. Instead, we model our problem as a decentralized Partially Observable MDP (Dec-POMDP) to capture both varying demand and limited sensing. In this formulation, the true system state is hidden from the agents. They receive limited observations instead.

Formally, a Dec-POMDP is defined by the tuple $\langle n, \mathcal{S}, \{\mathcal{A}\}_{i=1}^n, \mathcal{R}, \mathcal{T}, \{\mathcal{O}\}_{i=1}^n, \Omega, \gamma \rangle$, where n agents observe locally and jointly control n intersections in a network. Agents interact with the environment through corresponding action and observation factors: $\{\mathcal{A}\}_{i=1}^n$ and $\{\mathcal{O}\}_{i=1}^n$. The goal is to maximize the global reward $\mathcal{R}$. However, due to the problem complexity and sensing constraints, we rather optimize a surrogate target, the summation of local rewards, i.e., $\mathcal{R} = \sum_{i=1}^n \mathcal{R}_i$. Ω is the joint observation function that maps the true state to the observations associated with the agents. In this study, the model is designed to be:

- **States:** The true state $s \in \mathcal{S}$ contains all network information — vehicle positions, velocities, signal statuses, etc. — is statistically sufficient to describe the system dynamics, but is only partially revealed to any agent.
- **Observations:** Agent i observes $o_i \in \mathcal{O}_i$, which in our implementation includes vehicle and queue counts per lane and the current signal stage. We later augment o_i with additional features to enforce policy constraints and address partial observability.

- **Actions:** Rather than raw signal phases, each agent selects a *stage* — a pair of nonconflicting NEMA phases — at each decision step. If the chosen stage matches the current stage, the controller `EXTEND`s; otherwise it `CHANGE`s, inserting the appropriate yellow and all-red intervals. Our operation strictly obeys the dual-ring dual-barrier NEMA standard [10], which will be later formulated as policy constraints.
- **Rewards:** The local reward $r_i \in \mathcal{R}_i$ adopts the difference in cumulative delay proposed in [11].

B. Policy Constraints

Real-world deployment requires strict adherence to operational regulations. These rules further complicates the POMDP problem by specifying valid action sets on certain **states**. Although they can be effectively modeled by (hard) policy constraints, they inevitably require observation augmentation to help agents distinguish between available action sets from different traffic context, which further complicates the searching space. In this study, we consider operational regulations issued by the transportation agency in Toronto, Canada:

- **Stage transitioning sequence:** Follows NEMA standard and the specific definition per intersection.
- **Minimum and maximum stage lengths:** Enforce lower and upper bounds on green times. The lower bound adapts to the presence of pedestrians in real-time.
- **Cycle bound:** Although eMARLINs operate second-by-second, it supports to impose an upper bound on the cycle length to prevent prolonged green stages starving other movements.
- **Callable stage actuation:** Skippable stages become mandatory when on recall (e.g., vehicle detector or pedestrian call).
- **Dynamic Leading Pedestrian Interval (dynamic LPI, dLPI):** An LPI [12] provides pedestrians with a head start of 5 seconds to begin crossing the street before vehicles, enhancing pedestrian visibility and safety. dLPI automatically resolves the conflict between LPI and protected left-turn phases.

Additional observation components are introduced to accommodate these policy constraints, including the stage progressions to the min./max. constraints, cycle progression to the upper bound, and vehicle and pedestrian actuation calls. They expand the dimensionality of $\mathcal{O}_i$ and make good generalization a harder task.

C. The Out-Of-Distribution Issue

While our constrained Dec-POMDP formulation can, in principle, capture traffic dynamics under all real-world conditions, in practice no finite set of demand profiles fully represents those dynamics. An RL agent learns from the traffic patterns it encounters during training, yielding good performance within that distribution, but may fail when deployment conditions differ significantly. This discrepancy between training and deployment distributions is known as the Out-Of-Distribution (OOD) problem (or sim-to-real gap).

Conventional controllers face a similar issue. Pre-timed TSC systems divide the day into Time-of-Day (ToD) plans — each a "specialist" tuned to a particular traffic pattern — relying on the assumption of repeatable daily demand. Cycle-based adaptive systems (e.g., SCATS, SCOOT) adjust parameters like flow and occupancy at runtime but can struggle with sudden traffic shifts.

Our solution rests on two complementary strategies:

1) **Mitigating Partial Observability:** Agents must infer sufficient context to distinguish traffic states under limited sensing. We augment each agent's input by stacking the most recent H observations and sharing encoded histories with neighboring agents. A Transformer encoder processes these sequences, forming the core of the eMARLIN-Transformer algorithm and enhancing both local decision-making and inter-intersection coordination.
2) **Robust Training:** To ensure generalization, we expose the agent to a diverse spectrum of simulated scenarios — using domain randomization techniques such as demand scaling, mixture sampling, and stochastic perturbations — and apply early termination on "deadlock" states. This broad training regime transforms extrapolation challenges into interpolation tasks, yielding the eMARLIN-Robust approach.

Crucially, these two challenges are intertwined: without a powerful PO-mitigating encoder, the agent may fail to capture the distribution shift in training and struggle to learn even under robust training; without robust training, the same encoder will simply overfit to its limited training data. Addressing both is therefore necessary. The following section details the Transformer-based policy architecture and the robust training pipeline that together enable reliable, generalizable traffic signal control.

IV. METHODOLOGY

A. eMARLIN-Transformer

eMARLIN [13], [14] is a distributed multi-agent DQN algorithm designed to handle the complex, coupled dynamics of traffic-signal coordination. It features an efficient communication mechanism that shares each agent's local knowledge—encoded as compact embeddings—with its neighbors, thereby reducing communication bandwidth and costs.

Each agent consists of two components: an *Encoder* and an *Executor*. The Encoder compresses local observations into a low-dimensional embedding, minimizing the data transmitted to neighbors. The Executor then uses both observation embeddings of its own and those received from immediate neighbors to select actions. This architecture balances scalability and coordination: each agent trains its Encoder and Executor locally, avoiding network-wide backpropagation while ensuring efficient distributed learning.

To mitigate partial observability, we integrate a Transformer into the Encoder, yielding the eMARLIN-Transformer. We augment each agent's input with the past $H = 20$ seconds of observations and replace the original

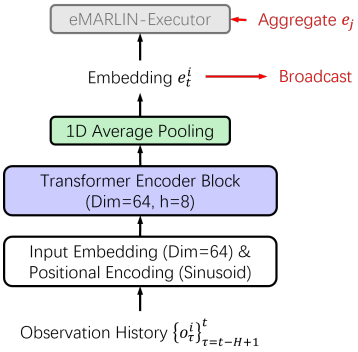


Fig. 2: The architecture of the Encoder module of eMARLIN-Transformer.

fully connected Encoder with a canonical Transformer encoder block (Figure 2). During training and inference, fixed-length observation sequences are processed via self-attention, enabling efficient parallel computation and improved temporal modeling.

Stabilizing learning in this Transformer-based RL setting requires careful architectural tuning [15], [16]. We select the depth and width of the standard encoder to balance capacity against the memory and latency constraints of real-time control. To leverage sequential data, we sample up to H past observations, reverse their order before padding and positional encoding, ensuring the most recent time step always occupies the same position. Because zeros in padded steps could confuse the model, we apply a padding mask within the self-attention layers and in the post-attention pooling layer. It prevents the model from attending to (or propagating gradients through) padded time steps, ensuring that the Transformer focuses strictly on valid traffic observations.

Positional encoding is important to Transformers because it injects information about the order of inputs into the model, enabling it to capture sequence structure despite having no inherent recurrence or convolution. Instead of using standard absolute positional encoding methods like sinusoidal mapping, we adopt the Rotary Position Embedding (RoPE) [17] in eMARLIN-Transformer. RoPE embeds relative positions directly into the attention mechanism by rotating the query and key vectors according to their positions, granting Transformers efficient relative position reasoning capability.

B. Robust Training

Having equipped our agents with this PO-mitigating architecture, we now introduce two key training techniques — early resignment and domain randomization — to improve efficiency and address the OOD challenge.

a) Early Resignment: In the RL pipeline, early stages of training often involve random or suboptimal decisions that can lead to network congestion and even deadlocks, especially due to the imperfect rule modeling in simulators. To mitigate the detrimental effects of such uninformative or harmful training episodes, we introduce an early resignment mechanism — a strategy commonly used in RL literature [18]. Instead of letting each episode run to a fixed horizon, early resignment terminates episodes prematurely when the agent enters a "bad" state identified through low value estimates or accumulated rewards, such as a deadlock. This approach accelerates learning by avoiding unproductive scenarios and improves training stability by reducing noisy gradients, making it particularly advantageous for value-based methods.

b) Domain Randomization: The core technique introduced in the eMARLIN-Robust approach to alleviate sim-to-real issue is domain randomization. This approach expands the variety of experiences the agent encounters while ensuring that the simulated conditions remain sufficiently close to reality.

This study introduces a set of domain randomization strategies to generalize demand patterns from a group of demand profiles identified from real-world data. For example, given 3 base demand profiles $\mathcal{D}_A, \mathcal{D}_B, \mathcal{D}_C$, each defined by a set of Origin-Destination (OD) matrices $\mathcal{D}_k = \{OD^k(t)\}_t, k = A, B, C; t = 0, 1, \ldots, T_k$, we apply the following strategies:

1) **Demand scaling:** Applies a scaling factor α to all OD matrices in a base demand, $OD^K(t) = \alpha \cdot OD^k(t), t = 0, 1, \ldots, T_k$, where α can be randomly selected in real-time or be prefixed. This strategy uniformly scales all traffics, keeping the spatial pattern unchanged while providing agents knowledge about over or under saturation conditions.
2) **Demand mixture:** Computes a mixture of all base demand profiles, $OD^K(t) = x_1 \cdot OD^A(t) + x_2 \cdot OD^B(t) + x_3 \cdot OD^C(t), t = 0, 1, \ldots, T_k$, and forcing $x_1 + x_2 + x_3 = 1$ to ensure that the mixture is interpolating the spatial pattern of base demand profiles. A mixture truncates the horizon to the shortest base demand profile, $T_K = min(T_A, T_B, T_C)$. These mixtures imitate fluctuation and the intermediate demand patterns occur in between different ToDs.
   We sample mixture coordinates (x_1, x_2, x_3) with the Dirichlet distribution. The Dirichlet distribution is a distribution over a probability simplex parameterized by a vector of concentration parameters $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3)$, where $\alpha_k > 0$:

$$f(\boldsymbol{x}; \boldsymbol{\alpha}) = \frac{1}{\mathrm{B}(\boldsymbol{\alpha})} \prod_{k=1}^{3} x_k^{\alpha_k - 1},$$

   where the normalizing constant is the multivariate beta function:

$$\mathrm{B}(\boldsymbol{\alpha}) = \frac{\prod_{k=1}^{3} \Gamma(\alpha_k)}{\Gamma(\sum_{k=1}^{3} \alpha_k)}.$$

   The concentration parameters $\boldsymbol{\alpha}$ controls in which area the samples will concentrate to. When all $\alpha_k = \alpha$, it is a symmetric Dirichlet. The choice $\boldsymbol{\alpha} = (1, 1, 1)$ recovers a uniform distribution, with smaller α encouraging sparsity (most of the values in a sample will be close to 0, concentrating around the vertices and edges of the simplex), with larger α pushing the concentration towards the center. In an asymmetric Dirichlet, the

distribution is more concentrated towards the vertex with greater α_k.

3) **Demand perturbation:** Randomly perturbs all OD pairs in all time intervals in a base demand profile, $OD_{ij}^K(t) = \alpha_{ij}(t) \cdot OD_{ij}^k(t), t = 0, 1, \ldots, T_k$, where $\alpha_{ij}(t) \sim \mathcal{N}(1, \sigma^2)$ are independently and identically distributed (i.i.d.). By adding subtle disturbances to every demand pair, this strategy introduces new spatial and temporal patterns, while maintaining an acceptable distance to the base demands. It attempts to simulate the subtle changes in traffic conditions from day to day.
4) **Pedestrian demand perturbation:** Uses the same vehicle demands, but perturbs the pedestrian arrival pattern. When a pedestrian cross operates on actuation mode, the presence of pedestrian significantly affects decision-making: it can not only activate a stage when there is no vehicle demand, but also elongate the minimum green constraint for non-skippable stages. Base demands assume a frequent service to pedestrian crossing; while this strategy samples pedestrian arrival from real-world collected pushing button activation logs.

Our fully parallelized training platform efficiently launches multiple rollout workers on separate simulator instances. Each rollout worker samples scenarios according to prescribed ratios for each randomization strategy. This integration of diverse domain randomization with efficient data collection yields the eMARLIN-Robust agent, which generalizes to a wide range of conditions and narrows the sim-to-real performance gap.

V. EXPERIMENTS

In collaboration with the City of Toronto, we conduct a pilot project targeting on in-field deployment on a busy region at the crossing of east–west Sheppard Avenue East and north–south Ontario Highway 404 (Hwy404) in Toronto, Ontario, Canada (Figure 3). Below, we present preliminary results from the simulated validation stage.

A. Test-bed

The chosen region exhibits highly complex and diverse traffic conditions: a busy transit terminal, an integrated shopping center, multiple highway ramps, and intersections with varied layouts. Traffic demand varies both temporally and spatially. During peak periods, vehicle, transit, and pedestrian volumes surge due to the proximity of highway ramps and transit hubs. Together, these factors create a challenging environment where the in-field reliability and the robust control paradigm of eMARLIN-Robust can be rigorously evaluated.

We focus control on four heavily congested intersections along Sheppard Avenue East. The simulation, built in Aimsun [19], covers 12 signalized intersections spaced 150–300 m apart. Demand profiles — derived from Ministry of Transportation Ontario traffic-volume data[1] — span three

[1]Ministry of Transportation Ontario: Traffic Volumes at Intersections for All Modes. https://open.toronto.ca/dataset/traffic-volumes-at-intersections-for-all-modes/

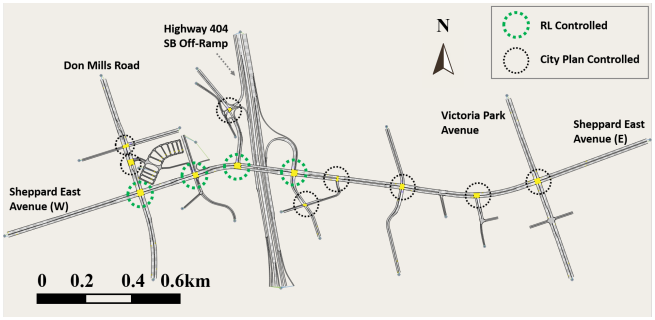


Fig. 3: The Sheppard-Hwy404 testbed.

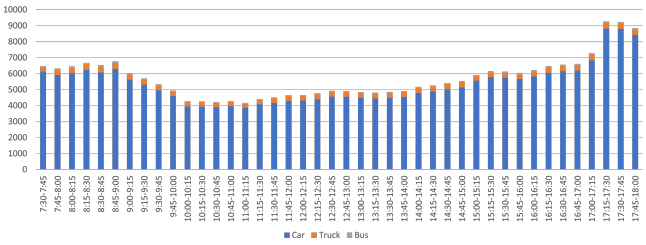


Fig. 4: The temporal distribution of the traffic demand in the Sheppard - Hwy404 testbed.

major ToD periods: the morning peak from 7:30 AM to 10:00 AM, the day-time off-peak from 10:00 AM to 3:00 PM, and the evening peak from 3:00 PM to 6:00 PM.

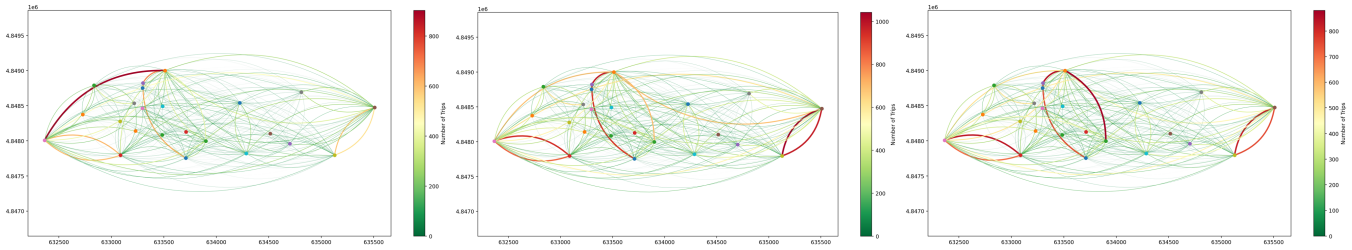
Vehicle types include cars, trucks, and buses following Toronto Transit Commission and York Region Transit schedules. We calibrate turning ratios using City of Toronto movement counts, achieving a strong linear fit ($R^2 = 0.9119$).

a) Temporal Demand Patterns: Figure 4 shows the characteristic dual-peak demand profile across the daytime, partitioned for each vehicle type.

b) Spatial Demand Patterns: The demand also shows strong spatial unevenness and its distribution varies significantly throughout a day. Figure 5 presents OD flows by ToD. Colored dots mark centroids that generate and attract vehicles, and colored arcs connecting ordered centroid pairs represent their demand volume: the more close to red, the higher. Note that we omit the dominant flow between the Hwy404 north and south ends as their volume dominates other OD pairs, preserving legibility of other OD pairs.

c) Intersection Layouts: From west to left, the four eMARLINs-controlled intersections are:

- **TCS0627:** Four approaches. All sides have pedestrian crossings operating on recall mode.
- **TCS1453:** Five approaches, four for all vehicles plus an extra dedicated for serving transit vehicles departing from the transit terminal located at the north-east corner of the TCS0627. Two NEMA phases are allocated to this transit approach, meaning it can be served up to twice upon request in a cycle. Only the north, west, and south sides have pedestrian crossings, with the north and south crossings operating on recall mode. The west side crossing is activated by pedestrian push button.



(a) AM peak (b) Daytime off-peak (c) PM peak

Fig. 5: The spatial distribution of the traffic demand and its variance in time.

- **TCS0746:** Three approaches. A single pedestrian crossing is located on the north side, operating on recall mode.
- **TCS0747:** Four approaches. Three pedestrian crossings are located on the north, west, and south sides. The west crossing is activated by push buttons, while the rests operate on recall mode. The northbound and southbound phases in this intersection are served non-simultaneously.

This testbed's complexity in spatial-temporal demand variability highlights the necessity of both robustness and strict policy constraint enforcement to achieve reliable, real-world performance.

B. Experiment Design

We compare eMARLIN variants against the City of Toronto's field-deployed TransSuite system — a semi-actuated, NEMA-based controller calibrated by traffic engineers using real-world data and operating with multiple time-of-day plans. In our Aimsun network, the four target intersections run under eMARLIN variants (honoring the policy constraints in Section III-B), while all other intersections remain under TransSuite control. We measure performance by total stop delay—the cumulative time that vehicles travel below 2 m/s within the controlled intersections.

During robust training, we expose eMARLIN-Robust to a diverse set of domain-randomized scenarios derived from three base profiles (AM, Off-peak, PM):

- **Demand scaling (AM_x1.1):** AM profile scaled by factor 1.1 (other scaling factors reserved for generalization tests).
- **Demand Mixture (DM):** Samples from a symmetric Dirichlet(0.5, 0.5, 0.5) over the three bases.
- **Demand Perturbation (GP):** Each base profile perturbed by independent $\mathcal{N}(1, 0.1^2)$ noise, yielding AM_GP, Off_GP, PM_GP.
- **Pedestrian demand Perturbation (PP):** Base profiles overlaid with day-specific pedestrian arrival patterns, yielding AM_PP, Off_PP, PM_PP.

These eight scenario types are sampled during training in the fixed ratio

$$\text{AM\_x1.1: DM: AM\_GP: Off\_GP: PM\_GP: AM\_PP: Off\_PP: PM\_PP} = 3:6:2:2:2:1:1:1.$$

To reduce evaluation overhead and avoid cherry-picking, we periodically evaluate the policy on the three base scenarios (AM, Off-peak, PM) during training. The checkpoint with the best average performance on these base profiles is selected as the final policy for all subsequent tests.

C. Robustness Evaluation Pipeline

We assess controller robustness across scenarios derived from modifications of three base demand profiles, mirroring the domain randomization strategies in Section IV-B. The evaluation pipeline comprises six tests and 75 distinct scenarios; each scenario is run with three random seeds to capture simulation stochasticity, for a total of 225 evaluation runs. Most test scenarios *differ* from those used in training. The tests are:

1) **Test 1 - base demand test:** Three scenarios, one for each base profile, establish baseline performance under standard conditions.
2) **Test 2 - demand scaling test:** Fifteen scenarios scaling each base profile by factors $\{0.5, 0.75, 0.9, 1.1, 1.25\}$. These scenarios provide an aggressive extrapolation from the base demands, testing the controller's generalizability under varied and scaled conditions.
3) **Test 3 - demand mixture test:** Nine scenarios defined by barycentric combinations of [AM, Off-peak, PM] with coordinates: [0.7, 0.3, 0], [0.7, 0, 0.3], [0.3, 0.7, 0], [0, 0.7, 0.3], [0.3, 0, 0.7], [0, 0.3, 0.7], [0.6, 0.2, 0.2], [0.2, 0.6, 0.2], and [0.2, 0.2, 0.6]. These scenarios interpolate between the base demand profiles.
4) **Test 4 - empirical pedestrian arrival pattern test:** Three scenarios applying real-world pedestrian arrival patterns to each base profile, extending evaluation to multi-modal conditions.
5) **Test 5 - demand perturbation test:** Fifteen scenarios in which each base profile is perturbed by Gaussian noise controlled by five prefixed random seeds, introducing spatial and temporal randomness.
6) **Test 6 - demand scaling with empirical pedestrian arrival pattern test:** Fifteen scenarios combining the scaled profiles of Test 2 with empirical pedestrian arrival data, further testing generalization under realistic multi-modal demand.
7) **Test 7 - demand perturbation with empirical pedestrian arrival pattern test:** Fifteen scenarios combining

the perturbed profiles of Test 5 with empirical pedestrian arrival data.

In summary, this pipeline challenges the controller across both interpolation (mixtures, perturbations) and extrapolation (scaling, pedestrian patterns) domains. By systematically varying demand characteristics, we rigorously evaluate generalizability and identify potential weaknesses, ensuring controller reliability in real-world urban deployments.

D. Results

1) eMARLIN-Specialist: We first train three eMARLIN-Specialist agents in pedestrian perturbed profiles corresponding to three base demands to demonstrate the near-oracle performance ATSC systems can reach. This testbed is highly constrained, leaving a minimal room for ATSC systems making improvements. Hence, on a certain scenario, specialist agents backboned with the original eMARLIN and eMARLIN-Transformer algorithms perform similarly. For simplicity, we present the specialist agents with eMARLIN backbone in this section.

Table I summarizes the performance of eMARLIN specialists across three scenarios. In each scenario, the corresponding specialist performs the best, presenting the outstanding delay time saving capability eMARLIN algorithms provide. However, when we apply specialists to the other two scenarios that they did not experience during training, they do not always perform ideally.

As shown in Figure 4, the heaviest demand in this network over the day occurs in the PM peak, from 5 PM to 6 PM. The PM peak specialist experienced saturated traffic in its training and hence can generalize to the AM peak scenario that also has heavy traffic. The PM peak specialist also experienced a under-saturation traffic during training PM and generalizes well to the off-peak period. The off-peak specialist, which only experienced light traffic in its training did not generalize well to peak demands. It even lost in the comparison to the TransSuite baseline in two peak ToDs, because it overfits to the light traffic scenario and can hardly adapt to heavy traffic. The AM peak specialist shows similar generalization capability to the PM specialist, as it also experienced saturate to light traffic during training.

However, even the AM and PM specialist can generalize to an extent, these three scenarios can only cover a limited variety of traffic patterns that the controllers may encounter in the real world. Real-world traffic can present hourly, daily, and seasonal trends, that can be hardly described by these three or a few more countable demand profiles. It is hard to guarantee that the AM peak specialist here can reliably generalize well across different traffic patterns after deployment. A robust agent is still urgently needed for facilitating safely and reliably operation in the field.

2) eMARLIN-Robust: Running a controller on the robustness evaluation pipeline produces results that vary considerably in scale. To facilitate meaningful comparisons, we report the related performance (in terms of the total stop delay) compared to the TransSuite baseline. Hence, the lower the reported number is, the better performance it has.

Applying our full eMARLIN-Transformer-Robust framework — which combines the eMARLIN-Transformer encoder and comprehensive domain randomization — yields consistent gains across all 75 evaluation scenarios (225 runs). Table II summarize performance relative to the TransSuite baseline.

It is evident that the eMARLIN-Robust agent outperforms the TransSuite baseline on every test, achieving reductions in total stop delay of $17.7\% \sim 27.8\%$. This superior performance validates our robust training strategies (Section IV-B) in equipping a single, unified policy to adapt across diverse traffic conditions.

Note that when evaluating the TransSuite baseline, we assumed that an expert dynamically assigns the most appropriate TransSuite timing plan based on the scenario's ToD, ensuring that TransSuite operates at its optimal performance level. Yet, even under these favorable conditions, the eMARLIN-Robust agent demonstrates a substantial performance advantage.

For ablation, we also evaluate the AM_PP eMARLIN-Specialist, together with an eMARLIN-Robust agent trained with the same robust training scheme. These two eMARLIN policies generally outperform or match TransSuite system across tests, demonstrating the strength of the underlying eMARLIN framework. However, on individual runs they sometimes fall behind TransSuite. In contrast, eMARLIN-Transformer-Robust is showing consistent improvement over TransSuite in every run and further reduces delay by an additional 12.2% on average compared to agents trained without the robust pipeline.

These significant improvements validate our strategic decision to adopt a unified, adaptive policy over multiple specialist plans. The agent's real-time responsiveness to dynamic traffic conditions without the need for plan-switching represents a major advancement in adaptive traffic signal control, ultimately contributing to smoother urban traffic flow and reduced overall delay.

3) Synergy of Transformer Encoding and Robust Training: Together, these results validate that both methodological contributions are indispensable:

Transformer integration is crucial for stabilizing learning under shifting distributions. By capturing temporal context and mitigating partial observability, it lays the foundation for any further generalization. *Robust training* via domain randomization expands the agent's experience beyond any one ToD, preventing overfitting. The powerful encoder needs to be properly trained to generalize well.

Only the combination embedding rich historical context through Transformers and exposing the agent to a wide variety of simulated conditions yields an adaptive, reliable policy fit for real-world deployment.

VI. CONCLUSION AND DISCUSSION

We presented eMARLIN-Transformer-Robust, a decentralized multi-agent framework that integrates Transformer-based history encoding with a robust training pipeline. We formally cast the problem as a Dec-POMDP, integrating

TABLE I: eMARLIN-specialist performance in different ToD scenarios compared to the TransSuite baseline (averaged across the evaluation results from three random seeds)

Total Stop Delay (s)	TransSuite	AM_PP eMARLIN-Specialist	Off_PP eMARLIN-Specialist	PM_PP eMARLIN-Specialist
AM_PP	1,721,720	**1,414,416**	2,164,870	1,552,620
Off_PP	2,387,020	1,451,807	**1,261,514**	1,308,350
PM_PP	2,148,860	1,989,048	2,342,670	**1,874,254**

TABLE II: eMARLIN-Robust performance in the robustness evaluation pipeline compared to the TransSuite baseline (averaged across the evaluation results from three random seeds). Numbers indicate normalized scores, the lower, the better.

Normalized Score	eMARLIN-AM-Specialist	eMARLIN-Robust	eMARLIN-Transformer-Robust
Test 1 - base demand test	0.898	0.763	**0.722**
Test 2 - demand scaling test	1.000	0.969	**0.817**
Test 3 - demand mixture test	0.898	0.917	**0.823**
Test 4 - empirical pedestrian arrival pattern test	0.795	0.796	**0.747**
Test 5 - demand perturbation test	0.844	0.873	**0.776**
Test 6 - demand scaling with empirical pedestrian arrival pattern test	0.877	0.870	**0.724**
Test 7 - demand perturbation with empirical pedestrian arrival pattern test	0.789	0.785	**0.744**
Averaged	0.877	0.872	**0.770**

hard policy constraints to ensure safety and regulatory compliance. Through an extensive simulated field test, we demonstrated that (1) Transformer encoding is essential for mitigating partial observability and stabilizing learning under shifting traffic dynamics, and (2) robust training techniques are critical to achieve strong generalization to unseen demand patterns. eMARLIN-Transformer-Robust consistently outperforms both ToD-specialists and the TransSuite baseline, adapting seamlessly without plan switching.

Next, we will deploy eMARLIN-Robust in live field trials and explore city-wide scaling and rare-event adaptation. By combining advanced sequence modeling with scenario-based training, eMARLIN-Robust charts a path toward more intelligent, resilient urban traffic control.

REFERENCES

[1] J. Saunders, "Transportation planning as infrastructural fix: Regulating traffi c congestion in the greater toronto and hamilton area," in *The Organization of Transport*. Routledge, 2014, pp. 207–227.

[2] M. Grote, I. Williams, J. Preston, and S. Kemp, "Including congestion effects in urban road traffic CO2 emissions modelling: Do local government authorities have the right options?" *Transportation Research Part D: Transport and Environment*, vol. 43, pp. 95–106, 2016.

[3] S. El-Tantawy, B. Abdulhai, and H. Abdelgawad, "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): methodology and large-scale application on downtown toronto," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1140–1150, 2013.

[4] X. Wang, B. Abdulhai, and S. Sanner, "A critical review of traffic signal control and a novel unified view of reinforcement learning and model predictive control approaches for adaptive traffic signal control," *Handbook on Artificial Intelligence and Transport*, pp. 482–532, 2023.

[5] J. Ault and G. Sharon, "Reinforcement learning benchmarks for traffic signal control," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[6] V. Jayawardana, C. Tang, S. Li, D. Suo, and C. Wu, "The impact of task underspecification in evaluating deep reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 23 881–23 893, 2022.

[7] J.-H. Cho, V. Jayawardana, S. Li, and C. Wu, "Model-based transfer learning for contextual reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 37, pp. 88 279–88 319, 2024.

[8] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[10] FHWA, *Traffic Signal Timing Manual*. U.S. Department of Transportation Federal Highway Administration, 2005, ch. Operational and Safety Analysis.

[11] S. M. A. Shabestary and B. Abdulhai, "Deep learning vs. discrete reinforcement learning for adaptive traffic signal control," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 286–293.

[12] S. Saneinejad and J. Lo, "Leading pedestrian interval: Assessment and implementation guidelines," *Transportation Research Record*, vol. 2519, no. 1, pp. 85–94, 2015.

[13] X. Wang, A. Taitler, I. Smirnov, S. Sanner, and B. Abdulhai, "eMARLIN: Distributed coordinated adaptive traffic signal control with topology-embedding propagation," *Transportation Research Record*, pp. 1–13, 2023.

[14] ——, "eMARLIN+: Addressing partial observability to promote traffic signal coordination by leveraging historical information," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 12, pp. 21 380–21 392, 2024.

[15] W. Li, H. Luo, Z. Lin, C. Zhang, Z. Lu, and D. Ye, "A survey on transformers in reinforcement learning," *arXiv preprint arXiv:2301.03044*, 2023.

[16] E. Parisotto, F. Song, J. Rae, R. Pascanu, C. Gulcehre, S. Jayakumar, M. Jaderberg, R. L. Kaufman, A. Clark, S. Noury *et al.*, "Stabilizing Transformers for reinforcement learning," in *International conference on machine learning*. PMLR, 2020, pp. 7487–7498.

[17] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu, "Roformer: Enhanced transformer with rotary position embedding," *Neurocomputing*, vol. 568, p. 127063, 2024.

[18] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[19] Aimsun, *Aimsun Next 22 User's Manual*, Aimsun Next 22.0.1 ed., Barcelona, Spain, Accessed on: May 20, 2023 2022. [Online]. Available: https://docs.aimsun.com/next/22.0.1/