Temporal Transfer Learning for Traffic Optimization with Coarse-Grained Advisory Autonomy

Jung-Hoon Cho, *Graduate Student Member, IEEE*, Sirui Li, Jeongyun Kim, and Cathy Wu, *Member, IEEE*

***Abstract*—The recent development of connected and automated vehicle (CAV) technologies has spurred investigations to optimize dense urban traffic, maximizing vehicle speed and throughput. This article explores *advisory autonomy*, in which real-time driving advisories are issued to human drivers, thus achieving near-term performance of automated vehicles. Due to the complexity of traffic systems, recent studies of coordinating CAVs have leveraged deep reinforcement learning (RL). Coarse-grained advisory is formalized as zero-order holds, and we consider a range of hold durations from 0.1 to 40 s. However, despite the similarity of the higher frequency tasks for CAVs, a direct application of deep RL fails to generalize to advisory autonomy tasks. To overcome this, we employ zero-shot transfer, training policies on a set of source tasks—specific traffic scenarios with designated hold durations—and then evaluating the efficacy of these policies on different target tasks. We introduce temporal transfer learning (TTL) algorithms to select source tasks for zero-shot transfer, systematically leveraging the temporal structure to solve the full range of tasks. TTL selects the most suitable source tasks to maximize the performance of the range of tasks. We validate our algorithms on diverse mixed-traffic scenarios, demonstrating that TTL more reliably solves the tasks than baselines. This article underscores the potential of coarse-grained advisory autonomy with TTL in traffic flow optimization.**

***Index Terms*—Deep learning in robotics and automation, intelligent transportation systems, learning and adaptive systems, transfer learning.**

I. Introduction

RECENT advancements in connected and automated vehicle (CAV) technologies have opened up new frontiers in addressing the challenges of urban traffic congestion and associated environmental problems. The growing urgency to mitigate traffic-related issues, buoyed by advances in autonomous vehicles (AVs) and machine learning, is pushing the boundaries of urban roadway autonomy. As the transportation sector progressively moves toward a fully autonomous paradigm, the spotlight is firmly on devising innovative methods for traffic flow optimization, targeting key outcomes, such as enhanced ecodriving, throughput maximization, and congestion reduction [1], [2].

This article highlights the significant role of *advisory autonomy*, an approach where automated systems provide real-time driving guidance to human drivers to integrate seamlessly with other traffic and achieve better traffic flow. The crux of our research lies in demonstrating how advisory autonomy can enable human-driven vehicles to emulate the system-level performance of automated vehicles (AVs), providing a viable, cost-effective alternative in the near term. The notion of *coarse-grained advisory autonomy* is formalized through the lens of coarse-grained zero-order holds [3]. With this coarse-grained advice, instead of instantaneous controls ([4], [5], [6]), vehicles are provided with guidance that persists for a particular duration, thereby addressing the intricacies of fluctuating hold duration. This is significant as human drivers, unlike AVs, may find it challenging to adhere to frequent and rapid control changes. Concurrently, work on robustness to human compliance errors (response delay, speed deviation) shows that advisory performance can degrade sharply without explicit handling [7].

In this work, our objective is to develop an algorithm that, given a traffic scenario, can determine whether guidance that human drivers could conceivably follow would achieve outcomes comparable to those of AVs. We concentrate on human compatibility for traffic optimization and the ability of human drivers to match corresponding system-level metrics (such as the average speed of all vehicles and the throughput) rather than achieve accurate maneuvers where AVs possess clear advantages, such as being able to react to abrupt braking without hesitation. This article subsumes and extends Sridhar et al.'s [3] work, which originally formulated the problem as piecewise-constant control for traffic optimization. This article further elaborates to include both acceleration and speed guidance and validates on different traffic networks.

Integrating this approach with reinforcement learning (RL) presents an elegant way forward, given RL's structured

Received 24 February 2025; revised 28 August 2025; accepted 7 October 2025. Date of publication 24 November 2025; date of current version 8 December 2025. This work was supported in part by the National Science Foundation (NSF) CAREER under Award #2239566, in part by the Kwanjeong Educational Foundation Ph.D. Scholarship Program, in part by the MIT Energy Initiative (MITEI) Mobility Systems Center, the Kwanjeong scholarship, in part by the NSF under Award 2149548, and in part by the MIT Amazon Science Hub. This article was recommended for publication by Associate Editor J. Le Ny and Editor J. Bohg upon evaluation of the reviewers' comments. *(Corresponding author: Jung-Hoon Cho.)*

Jung-Hoon Cho is with the Department of Civil and Environmental Engineering and the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: jhooncho@mit.edu).

Sirui Li is with the Institute for Data, Systems, and Society and the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: siruil@mit.edu).

Jeongyun Kim is with the Department of Mechanical and Automotive Engineering, Seoul National University of Science and Technology, Seoul 01811, South Korea (e-mail: jkim@seoultech.ac.kr).

Cathy Wu is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139 USA, also with the Institute for Data, Systems, and Society, Massachusetts Institute of Technology, Cambridge, MA 02139 USA, and also with the Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: cathywu@mit.edu).

Digital Object Identifier 10.1109/TRO.2025.3636819

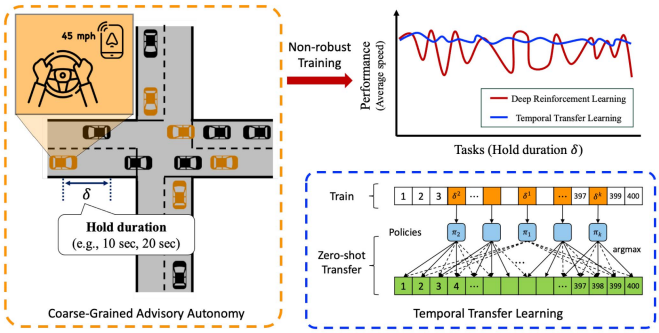


Fig. 1. Illustrative figure of TTL for the coarse-grained advisory system. In a *coarse-grained advisory system*, vehicles receive persistent guidance for a specified hold duration rather than instantaneous controls. The system performance of this system shows the nonrobustness to the hold duration of deep RL when trained exhaustively. We propose TTL methods designed to select source training tasks based on temporal features. In comparison to the exhaustive and multitask training methods, TTL provides an intermediate number of policies to train to solve a full set of tasks.

framework for sequential decision-making. While deep RL has emerged as a potent tool for this purpose, its direct application to the advisory system has exposed a degree of instability, characterized by jagged performance over a range of tasks, echoing the findings of Sridhar et al. [3]. This inconsistency and brittleness necessitate a more sophisticated approach to deep RL, one that can more reliably handle the complexities of real-world traffic scenarios encountered by advisory systems.

To confront these challenges head-on, we turned to transfer learning, a widely employed technique in numerous research fields that enables the utilization of knowledge acquired from one task to enhance performance in another related task [8], [9]. Specifically, transfer learning can be applied to adapt the pretrained policy to a new task or to initialize a learning algorithm with pre-existing knowledge, substantially expediting the learning process and boosting overall performance. Transfer learning has been successfully applied to improve the efficiency and training performance of traffic management systems [4], [10], [11]. In this context, we employ zero-shot transfer, where policies are trained on a source task–specific traffic scenarios with designated hold duration—and then evaluated on different target tasks. This approach is computationally efficient as it obviates the need for any additional fine-tuning, directly leveraging the trained policies to new scenarios.

We introduce two temporal transfer learning (TTL) algorithms– greedy temporal transfer learning (GTTL) and coarse-to-fine temporal transfer learning (CTTL). A high-level overview of the proposed TTL framework for coarse-grained advisory autonomy is shown in Fig. 1. These algorithms adeptly leverage the temporal similarities across tasks to judiciously select training tasks, thereby significantly facilitating the training efficiency and overall performance. The essence of TTL lies in its capability to seamlessly transfer knowledge acquired from one task to another, circumventing the often observed training brittleness in deep RL algorithms. The TTL approach provides a structured advantage by systematically leveraging temporal structures inherent in the task domain. Our GTTL methods especially leverage a linear generalization gap, which allows better estimation of zero-shot generalization across different tasks. This ability of TTL to draw insights from prior models offers a promising avenue to circumvent the fragility often observed in deep RL training. Then, to evaluate our algorithm's generalizability, we consider validation on various traffic scenarios in which mixed-autonomy traffic has been proven effective for traffic optimization [5], [6].

The core contributions of this article are twofold as follows.

- We delve into a *coarse-grained advisory*, presenting a compelling case for its viability in enhancing system-level traffic outcomes. Our empirical evidence underscores the possibility of furnishing human drivers with guidance that mirrors AV behavior, leading to tangible traffic improvements. Such findings pave the way for considering human drivers as immediate, practical alternatives to full-fledged AV deployments.
- Our research introduces TTL algorithms, a robust methodology specifically designed to tackle the training brittleness intrinsic to deep RL algorithms. TTL can be promising in evolving generalizable training paradigms for complex traffic optimization tasks by adeptly identifying sources of variation and harnessing insights from pre-existing models.

II. Related Work

A. RL for Mixed Autonomy Traffic

As we await the era of fully automated vehicles, we can anticipate a mixed autonomy system where automated and human-driven vehicles share the road. In such a system, controlling only a small proportion of the vehicles can significantly improve the overall traffic flow [5], [10]. Several studies have explored the potential of RL in addressing the challenges posed by the coexistence of AVs and human-driven vehicles. Researchers have worked on enhancing traffic efficiency in mixed autonomy settings using deep RL-based approaches, showing that it can eliminate stop-and-go traffic and mitigate congestion [2], [4], [5], [6], [10], [12]. These studies collectively highlight the potential of RL in optimizing mixed autonomy traffic, paving the way for enhanced safety, efficiency, and performance in transportation systems.

B. Advisory Autonomy

Advisory systems in roadway autonomy span a broad range of applications, from enhancing safety to mitigating traffic congestion. These systems provide considerable benefits to users. For instance, collision warning alerts have been employed to ensure the driver's safety [13], and speed advisory systems at signalized intersections help users pass the green light efficiently [14]. At a system level, on the other hand, the advisory system provides system-level traffic optimization. For example, speed advisory systems contribute significantly towards ecodriving [15] and personalized advisory systems have been introduced to mitigate traffic congestion [16], [17]. Furthermore, roadway signs suggesting advisory speeds represent another form of advisory autonomy.

However, the application of advisory systems poses unique challenges given their interaction with human drivers. While fully automated vehicles can operate within clearly defined

parameters and constraints, human drivers exhibit different behaviors. For instance, as noted by Mok et al. [18], humans require a minimum of 5–8 s to appropriately transition control. This finding highlights the importance of considering the distinct characteristics and limitations of human drivers when developing control methods. For instance, Sridhar and Wu [3] identified two key characteristics of human-compatible driving policies: a simple action space and the capacity to maintain the same action for a few seconds. An example of a human-compatible advisory system is a coarse-grained control system, which is provably stable in the context of Lyapunov in mitigating congestion on single-lane ring roads [19]. This system, known as an action-persistent Markov decision process (MDP), successfully addresses the human need for simplicity and persistent actions. In light of these considerations, it is crucial to integrate human driving characteristics into the design of control methods for human drivers. Complementary to our effort in developing the transfer learning-based algorithm, Kim et al. [7] relaxed the perfect-compliance assumption by modeling driver delays and speed deviations, revealing substantial degradation without robustness measures and proposing RL-based advisories resilient to such errors.

C. Action Persistent MDPs and RL

In exploring action repetition within RL, the concept of semi-MDPs offers a rich framework for incorporating temporally abstract actions into the traditional MDP paradigm [20]. The ability to apply the same action across extended time periods allows for a simplified control strategy that can be beneficial for complex control problems.

Various methodologies such as reducing control granularity, implementing a skip policy, and applying temporal abstraction have been employed to analyze action repetition [21], [22], [23], [24]. Metelli et al. [24] introduced action persistence and the persistent fitted Q-iteration algorithm to modify control frequencies and learn optimal value functions. Lee et al. [25] addressed the multiple control frequency problems that guarantee convergence to an optimal solution and outperform baselines. In the context of transportation systems, Sridhar and Wu [3] investigated the use of piecewise constant policies for traffic congestion mitigation, providing a structured approach to guide human drivers in real-time.

These contributions collectively underscore the significance of action repetition and the strategic choice of control frequencies in RL. They also highlight the potential for translating these concepts into tangible traffic management solutions, exemplifying the intersection between theoretical research and practical application.

D. Transfer Learning in RL

Transfer learning is a popular technique used in various research domains to leverage the knowledge gained from one task to improve performance in another related task [8], [9]. In particular, transfer learning can be used to adapt a pretrained policy to a new task or to initialize a learning algorithm with pre-existing knowledge, thereby greatly accelerating the learning process and improving overall performance. In contrast to multitask learning's simultaneous approach, transfer learning applies knowledge from source tasks to optimize a particular target task, underscoring an asymmetrical relationship between tasks [9].

Transfer learning offers the advantage of significantly decreasing the amount of data needed for learning compared to traditional independent learning methods [26]. Dynamic transfer learning maps for multirobot systems can be obtained from the basic system properties of approximated physical models or experiments [27]. Kouw and Loog [28] not only delved into the specific instances and various techniques of domain adaptation but also highlighted the challenges of sequential domain adaptation. Moreover, transfer learning also has its benefits, as it requires a reduced number of data or training for new tasks, stemming from the shared representation of related tasks [26], [29], [30].

In robotics, transfer learning has been utilized for a wide range of applications such as robot manipulation, locomotion, and control [31], [32]. In the context of traffic settings, transfer learning has been applied to improve the efficiency and training performance of traffic management systems [4], [10], [11]. For example, Kreidieh et al. [10] proposed a transfer learning framework that can help the warm start for training policies to dissipate shockwaves from closed traffic scenarios to more complex open ones. Similarly, zero-shot policy transfer to adapt a pretrained policy for autonomous driving in a structured environment to an unstructured environment results in improved performance and safety [11].

Also, the transferability of the learned policies may differ at different levels of tasks; for instance, policies derived from more structured and informative tasks are more robust to diverse tasks [33]. Yan et al. [4] proposed a unified framework for traffic signal control using transfer learning to transfer knowledge across different intersections and adapt to varying traffic conditions. Also, transfer learning is used for real-time crash prediction [34], and traffic flow prediction in data-sparse regions [35].

RL-based methods require generating significant amounts of simulation data, which can be costly. However, transfer learning offers a solution to alleviate the burden of data generation and simulation for training each model. By employing an efficient training scheme, the model can quickly learn when, what, and where to transfer knowledge in scenarios with limited data availability [36]. The selection of source tasks is critical in transfer learning as it sets the foundation for the efficacy of knowledge transfer. Contextual relevance in source task selection is critical for the efficacy of the transferability of a policy, which may be predicted through its relation to the target task's characteristics [37], [38]. Furthermore, Agostinelli et al. [39] explored metrics that predict the success of transferred knowledge, facilitating the selection of source model ensembles to maximize performance on the target task. In addition to selecting individual source tasks, multitask learning can also be used to solve multiple related tasks [40]. Closest to our setting, Cho et al. [41] introduced model-based transfer learning, which explicitly models 1) the training performance via Gaussian processes and 2) a linear generalization gap over context, and then uses Bayesian optimization to pick source tasks with sublinear regret

guarantees. In contrast, our TTL specializes in the temporal context (hold duration δ) and yields simpler closed-form greedy and coarse-to-fine selection rules with area-coverage guarantees.

A hierarchical approach to task granularity can be beneficial as it allows for the refinement of coarse attributes while learning finer tasks. This method has been successfully employed by Wei et al. [42] in their work on vehicle reidentification tasks and in large-scale fault diagnosis tasks [43]. Our method leverages temporal locality in hold duration—transferring between nearby δ—which plays an analogous role to curriculum learning while explicitly optimizing source-task selection under a fixed training budget.

Overall, transfer learning has shown promising results in improving the efficiency and safety of traffic management systems by leveraging the similar temporal structure of a series of tasks and prior knowledge from related tasks.

III. Coarse-Grained Control

A. Coarse-Grained Guidance in Advisory Autonomy

Advisory autonomy stands for the automated system that provides guidance to human drivers rather than a fully controllable process. In this context, it is designed to work in the presence of human-driven vehicles, ensuring that controlled vehicles operate in a manner that is safe, predictable, and intuitive for human drivers. *Coarse-grained control* refers to the vehicle control system that gives control periodically. Coarse-grained control involves applying the same action to an AV for a fixed time segment. As we discussed in Section II-C, coarse-grained control can be interpreted as action-persistent MDPs with different control granularities.

B. Action Persistent MDPs

We consider N vehicles and assume that all vehicles are human-driven vehicles. A subset of these vehicles, defined by the fraction ρ, receives periodic guidance from the advisory system and is termed *guided vehicles*. The remaining vehicles, constituting the fraction $(1-\rho)$, are designated as *default-driven vehicles* and do not receive such guidance. Guided vehicles are human-driven vehicles with periodic assistance from a trained policy for coarse-grained control, $\pi(s_{t_m})$. The policy is applied at intervals $t_m = \delta m$, where $m \in \mathbb{N}_0$ ($\mathbb{N}_0$ as a set of nonnegative integers) and δ denotes the guidance hold duration. It is essential that the guidance hold duration is significantly shorter than the total horizon, denoted as ($H \gg \delta$). These vehicles receive guidance for any time t that falls within the range $[t_m, t_{m+1}]$. This action persistent MDP can be represented by the seven-tuple $\mathcal{M}_\delta = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, H, \gamma, \delta)$, where $\mathcal{S}$ defines the state space, $\mathcal{A}$ is action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is a transition probability distribution, $\mathcal{R}$ is reward function, H is a total time horizon, and γ is a discount factor. The transition probability function $P(s'|s,a)$ specifies the probability of transitioning to a state s' from a state s by taking action a. An agent's objective in an MDP is to find a policy π that maximizes the expected sum of rewards obtained over time, given the current state s and the actions it can take.

Coarse-grained control, also known as piecewise constant control or zero-order hold control, refers to the application of the same control action over a specified time segment length [3]. In other words, the same action determined at time step t_m is applied to the time segment of $t \in [t_m, t_{m+1}]$. In the single-lane ring, the simulation experiments reported that the hold duration could be extended to 24 s without degradation of the system performance [3]. This piecewise constant control is backed up with the simulator experiments to evaluate the effect of the coarse-grained advisory [44]. Hasan et al. [16], [17] also introduced a cooperative advisory system that leverages a novel driver trait conditioned personalized residual policy to guide drivers in ways that reduce traffic congestion.

C. Guidance Type

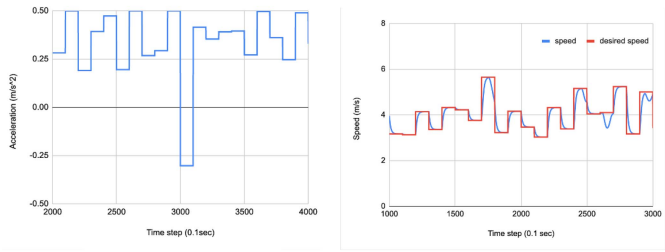
In our setting, we assume that the guided vehicle i observes the space headway $s_i(t)$ between ith vehicle and the preceding vehicle and its derivative $\dot{s}_i(t)$ can be computed from successive headway measurements or velocity differences when the preceding vehicle's velocity can be observed. Likewise, the velocity $v_i(t)$ comes from speed sensors in the vehicle. Based on the observation of the drivers, the coarse-grained controller provides the drivers with either the target raw acceleration or target speed.

Acceleration guidance: Acceleration guidance in advisory autonomy systems directs human drivers by recommending the optimal acceleration action from a continuous action set, determined by the trained policy π. Concretely, the policy π takes the observed traffic state at time t_m—namely $\big(s_i(t_m), \dot{s}_i(t_m), v_i(t_m)\big)$—and outputs a recommended acceleration for guided vehicle i for the duration $t \in [t_m,\ t_{m+1} = t_m + \delta]$.

For this acceleration guidance in the single-lane ring, Lyapunov analysis gives sufficient conditions for the stability of the coarse-grained advisory [19]. Moreover, the success of this guidance system critically depends on the interface design through which advisories are communicated. A few challenges are possible discomfort for drivers, the complexity of human drivers in accurately interpreting and executing precise acceleration commands, and an increased risk of manual execution errors.

Speed guidance: In contrast to acceleration guidance, speed guidance presents human drivers with a target speed that they should attain and maintain over the hold duration interval. This approach is motivated by the observation that many drivers find it more intuitive to adjust to a specific speed target rather than precisely following recommended accelerations [45]. Moreover, using acceleration guidance type for the coarse-grained control often struggles to achieve and sustain the optimal velocity as discussed in [46] and [47]. Accordingly, our trained policy π generates a recommended target speed for guided vehicle i based on its current state $\big(s_i(t_m), \dot{s}_i(t_m), v_i(t_m)\big)$. Once this target speed is communicated, the driver accelerates or decelerates to reach it as quickly and smoothly as possible, subject to comfort and safety constraints.

Researchers have worked on the speed advisory system [48], [49], [50]. For example, Liang et al. [48] guided the driver with



(a) (b)

Fig. 2. Two types of advisory system to the human drivers: acceleration guidance (a), speed guidance (b).

the speed for signal phase and timing in CAV environment. Wang et al. [51] reported that the human–machine interface displaying the difference between current and suggested speeds with the cooperative driving simulator improved the performance while displaying time difference harmed the speed adaptation. However, there are some drawbacks that human drivers tend to perceive the target speed as the easily broken speed limit and easily exceed it [52].

Fig. 2 intuitively depicts two distinct forms of advisory provided to drivers: acceleration and speed guidance. From a stabilization standpoint, speed guidance has certain advantages, as it enables the vehicle to maintain a constant speed throughout the hold duration. However, under acceleration guidance, the vehicle's speed is subject to change unless the acceleration is precisely zero. This inherent difference between the two forms of guidance gives rise to distinct behaviors and responses within the traffic system, as demonstrated in our results.

IV. Temporal Transfer Learning

In advisory autonomy, we guide human drivers by providing a predetermined period, known as the hold duration, indicating how long they should maintain their guided actions. We consider solving families of MDP tasks whose only difference is the guidance hold duration, since the control of humans can vary. Throughout, we use "task" in the transfer-learning sense to denote the same control problem evaluated at a particular guidance hold duration δ. Apart from δ, all environment and model parameters, such as the number of agents and road networks, are identical. Even in this setting, we find that RL will train successfully in some scenarios and unsuccessfully in others, with no clear pattern among the tasks. Similar findings have been documented in [53].

The algorithm introduced in this section is inspired by the intuition that an optimal strategy for hold duration δ should not differ significantly from that for hold duration $\delta' \approx \delta$. In particular, small perturbations of the hold duration should yield only minor changes to the action sequence and the closed-loop dynamics. Empirically, however, $J^*(\delta)$ and the learned policies exhibit a jagged, nonsmooth dependence on δ—a sign of training brittleness—which motivates zero-shot transfer across nearby tasks.

We aim to investigate multiple tasks to comprehend the intricacies of the coarse-grained advisory and its effectiveness in optimizing traffic flow, particularly in mitigating congestion. Addressing multiple tasks provides a holistic understanding of the system's behavior under various scenarios, thereby informing more robust optimization strategies. While solving multiple tasks simultaneously with a separate model per each task could be computationally intensive and resource-demanding, leveraging a pre-trained model and transfer learning for our specific tasks can drastically reduce the computational burden. The list of variables and functions used throughout this article is summarized in the Table I, ensuring clarity and ease of reference for the reader.

TABLE I
Table of Notations

Symbol	Description
δ	Guidance hold duration
$J(\delta)$	The performance of task with duration δ
A	The aggregate performance across different durations
k	Sequential step in source task selection
δ^k	The guidance hold duration chosen at kth step
π_k	The policy trained at the task at δ^k
$J^{\pi_k}(\delta^k)$	The performance of policy π_k evaluated at δ^k
$\Delta J(\delta_S, \delta_T)$	The generalization gap when the policy trained with δ_S transferred to the task with δ_T
$J_k(\delta)$	The performance updated with the best-performing performance among previously trained policies
S_k	A set of selected source tasks

A. Problem Definitions

Guidance hold duration and performance: In coarse-grained advisory settings, let $J(\delta)$ denote the performance metric for a task with a guidance hold duration of δ. For a coarse-grained policy π, we write this as $J^\pi(\delta)$. The hold duration δ spans from $\delta_{\min}$ to $\delta_{\max}$. We define the *aggregate performance* over the interval $[\delta_{\min}, \delta_{\max}]$ as the integral of $J(\delta)$

$$A(\delta_{\min}, \delta_{\max}) = \int_{\delta_{\min}}^{\delta_{\max}} J(\delta)\, \mathrm{d}\delta \tag{1}$$

which measures how the system performs across all hold durations. In practice, we approximate this integral via a discrete sum

$$\tilde{A}(\delta_{\min}, \delta_{\max}) = \sum_{\delta=\delta_{\min}}^{\delta_{\max}} J(\delta). \tag{2}$$

For notational simplicity, we will use $A(\delta_{\min}, \delta_{\max})$ or simply A to represent this aggregate performance.

Sequential source tasks selection problem: We next define the *sequential source tasks selection problem*, where the goal is to iteratively choose which hold duration δ^k to train on so as to maximize overall performance. We let K denote the transfer budget (i.e., the maximum number of source tasks we can train). At each iteration $k \in \{1, \ldots, K\}$, we train a policy on the task withhold duration δ^k. The set of selected source tasks at iteration k is denoted by S_k.

We write $J_k(\delta)$ for the estimated performance on a task withhold duration δ after k iterations of training on selected tasks. Training at δ^k produces a policy π_k with performance $J^{\pi_k}(\delta^k)$ on that same task. In practice, a policy trained on one task δ_S

may perform suboptimally when applied to a different task δ_T, due to a *generalization gap*. This gap captures how performance degrades when zero-shot transferring a policy from δ_S to δ_T. Such performance degradation has been studied in [54] and [55], and similar effects have been observed in multiobjective contexts [56]. The generalization gap is crucial for understanding the limits of transfer and for guiding the iterative improvement of task-specific policies in RL.

Definition 1. (Generalization gap $\Delta J(\delta_S, \delta_T)$): For a policy trained on source task δ_S and evaluated on target task δ_T, we define the *generalization gap* as the difference in performance $\Delta J(\delta_S, \delta_T) = J^{\pi_S}(\delta_S) - J^{\pi_S}(\delta_T)$.

We initialize the estimated performance $J_0(\delta)$ for all guidance hold durations δ within the range of interest to zero, which sets the baseline for subsequent improvements

$$J_0(\delta) = 0 \quad \forall \delta \in [\delta_{\min}, \delta_{\max}]. \tag{3}$$

After each iteration k, $J_k(\delta)$ is updated by taking the maximum over the best known policy so far and the newly trained policy (accounting for the generalization gap)

$$J_k(\delta) = \begin{cases} J^{\pi_k}(\delta^k), & \text{if } \delta = \delta^k \\ \max\big(J_{k-1}(\delta),\ J^{\pi_k}(\delta^k) - \Delta J(\delta^k, \delta)\big), & \text{otherwise.} \end{cases} \tag{4}$$

Definition 2. (Sequential source tasks selection problem): The problem is to select a sequence of tasks $\{\delta^1, \delta^2, \ldots, \delta^K\}$ that maximizes the aggregate performance. At iteration k, we pick δ^k to maximize $A_k(\delta^k)$, where

$$A_k(\delta^k) = \int_{\delta_{\min}}^{\delta_{\max}} \max\big(J_{k-1}(\delta),\ J^{\pi_k}(\delta^k) - \Delta J(\delta^k, \delta)\big)\, \mathrm{d}\delta. \tag{5}$$

The selection process continues for up to K iterations, thereby incrementally improving policy performance across all hold durations.

A conceptual illustration of this process is shown in Fig. 3. At each step, the newly trained policy may enhance performance in a neighborhood of δ^k, subject to the generalization gap when evaluated on other tasks.

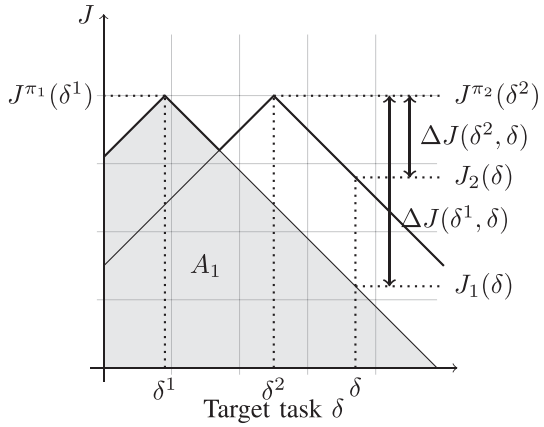


Fig. 3. Visualization of sequential source task selection and corresponding performance evaluations within the guidance hold duration space. The shaded region represents the aggregate performance A_1 after selecting δ^1 in the first step. The generalization gap $\Delta J(\delta^1, \delta)$ quantifies the performance drop when applying the policy trained at δ^1 to a target task with δ. At the second step, the selection of δ^2 updates the estimated performance of task with duration of δ from $J_1(\delta)$ to $J_2(\delta)$.

B. Modeling Assumptions

We now list the main assumptions used to formalize and analyze the source tasks selection problem in the context of TTL.

Assumption 1 (Constant upper bound performance): For any hold duration δ in $[\delta_{\min}, \delta_{\max}]$, the upper bound of the performance $J^*(\delta)$ is constant, denoted by J^*. Formally

$$J^*(\delta) = J^*, \quad \forall\, \delta \in [\delta_{\min}, \delta_{\max}]. \tag{6}$$

Assumption 1 is supported by empirical analysis within coarse-grained advisory autonomy settings, suggesting that various coarse-grained guidance tasks may uphold the same training performance. Our observations in single-lane ring environments validate this assumption [see Fig. 9(a)], although it is noted that in more complex scenarios, such as highway ramps, the training performance may decline as hold duration increases [see Fig. 9(b)].

Assumption 2 (Deterministic training performance): For any task trained withhold duration δ^k, training attains performance $J^{\pi_k}(\delta^k) \approx J^*(\delta^k)$; analysis uses $J^{\pi_k}(\delta^k) = J^*(\delta^k)$.

From Assumptions 1 and 2, training on any chosen δ^k achieves J^*, which simplifies the analysis of subsequent transfers.

Assumption 3 (Linear generalization gap): The generalization gap $\Delta J(\delta_S, \delta_T)$ between tasks δ_S and δ_T is linearly proportional to $|\delta_S - \delta_T|$. Specifically

$$\Delta J(\delta_S, \delta_T) = \begin{cases} \theta_L(\delta_S - \delta_T), & \text{if } \delta_S > \delta_T \\ \theta_R(\delta_T - \delta_S), & \text{otherwise} \end{cases} \tag{7}$$

where θ_L signifies the slope of transfer performance when transitioning from a coarser to a finer task, implying that $\delta_S > \delta_T$. Conversely, θ_R represents the slope when shifting from a finer to a coarser task, suggesting that $\delta_S < \delta_T$.

Assumption 4 (Symmetric generalization gap function): For simplicity, we assume the transfer slopes are equal, i.e., $\theta_L = \theta_R = \theta$.

Assumption 4 simplifies the analysis by asserting that the granularity of the task does not influence the rate of performance degradation during policy transfer.

Assumption 5 (Bounded slope of generalization gap function): We require that $J^* \geq \theta(\delta_{\max} - \delta_{\min})$, so that

$$\theta \leq \frac{J^*}{\delta_{\max} - \delta_{\min}}. \tag{8}$$

This bound ensures the generalization gap does not exceed the maximum possible performance, making geometric analyses tractable. If J^* is larger than $\theta(\delta_{\max} - \delta_{\min})$, the transfer from any point would be able to encompass the additional volume. Thus, without loss of generality, we can assume $J^* = \theta(\delta_{\max} - \delta_{\min})$ in our geometric analysis. If J^* is less than $\theta(\delta_{\max} - \delta_{\min})$, indicating a relatively constrained effective transfer range, the

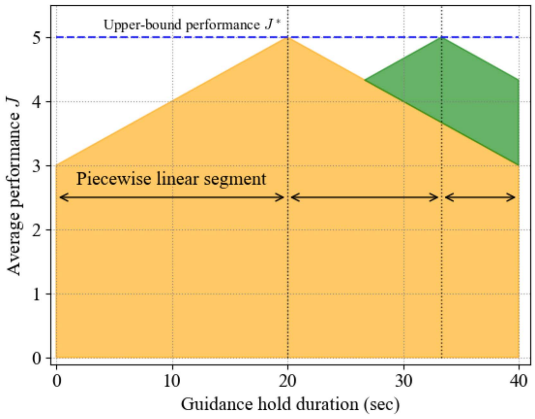


Fig. 4. Exemplified representation of the TTL process for source task selection. The graphic showcases the stepwise procedure for two iterations ($k = 2$), resulting in two segments demarcated by inflection points at δ^1 and δ^2. The upper bound performance J^* is indicated by the blue dotted line, as posited in Assumption 1, while the piecewise linear segments and their slopes, as governed by Assumptions 3 and 4, guide the selection of the next hold duration δ^k that will maximize the aggregate performance A_k. Each segment is assessed for its potential marginal contribution to A_k, with decisions influenced by the shape of the performance function $J(\delta)$, here visualized as transitions from the orange to the green area, signifying the shift in guidance hold duration from $\delta^1 = 20$ to $\delta^2 = 33.33$.

advantage of using transfer learning might be limited. This is due to the fact that TTL tends to benefit from the case where the amount of the generalization gap is prominent.

C. Optimal Strategy for Source Tasks Selection Problem

With several assumptions we made in the previous section, we can devise a systematic algorithm to solve the source tasks selection problem and choose the subsequent training source task based on simple geometry. We consider an analysis that simplifies the marginal performance improvement after each iteration to obtain intuition and provide a theoretical grounding for the TTL process.

As shown in Fig. 3 and supported by Assumptions 1–3, training on a selected source task yields optimal performance at that task, creating two distinct segments in the performance function. These segments, delineated by the selected task, can be modeled as piecewise linear functions. After $(k-1)$ iterations, there are k segments with inflection points at $\delta^1, \ldots, \delta^{k-1}$. Our objective is to select the next hold duration δ^k that maximizes the aggregate performance A_k. To this end, we evaluate each piecewise linear segment and compute its potential marginal increase in A_k, which depends on the shape of $J(\delta)$.

Figure 4 illustrates various decision rules for selecting the transfer point based on the shape of $J(\delta)$. Under Assumption 1, the performance upper bound J^* is depicted as a flat blue dotted line, while Assumptions 3 and 4 ensure that the transfer performance functions in both directions are linear with the same slope.

We formalize a greedy approach in theorem 1, which chooses δ^k to maximize the marginal increase in A_k within each piecewise linear segment. Although "greedy" focuses on one step at a time, it yields an efficient and interpretable selection rule under our linear-gap assumptions.

Algorithm 1: Greedy Temporal Transfer Learning (GTTL).

Input: MDP $\mathcal{M}_\delta$, Hold-duration range $[\delta_{\min}, \delta_{\max}]$, Upper bound area A^*, Termination criteria ε, Transfer budget K
Output: J_k and S_k
Initialize : $J_0(\delta) = 0\ \forall \delta \in [\delta_{\min}, \delta_{\max}]$, $A_k = 0$, $S_k = \{\}$, $\boldsymbol{\pi} = \{\}$, $k = 0$
while $(A_k \leq (1-\varepsilon)A^*)$ and $(k \leq K)$ **do**
  $\delta^{k+1} \leftarrow$ $\mathbf{FindGreedyTransferPoint}(S_k, J_k, \delta_{\min}, \delta_{\max})$
  $S_{k+1} \leftarrow S_k \cup \{\delta^{k+1}\}$
  $\pi_{k+1} \leftarrow \text{Train}(\mathcal{M}(\delta^{k+1}))$
  $\boldsymbol{\pi} \leftarrow \boldsymbol{\pi} \cup \{\pi_{k+1}\}$
  $J_{k+1}(\delta^{k+1}) \leftarrow J^{\pi_{k+1}}(\delta^{k+1})$
  $J_{k+1}(\delta) \leftarrow \max(J_k(\delta), J_{k+1}(\delta^{k+1}) - \Delta J(\delta^{k+1}, \delta))\forall \delta \in (\delta_{\min}, \delta_{\max}) \backslash \delta^{k+1}$
  $A_{k+1} = \int_{\delta_{\min}}^{\delta_{\max}} J_{k+1}(\delta)\mathrm{d}\delta$
  $k \leftarrow k+1$
end while
return J_k and S_k

Theorem 1: [Optimal source task selection for greedy transfer]: Consider a piecewise linear segment $[\delta_L, \delta_R]$ of $J_k(\delta)$. To maximize the marginal increase in A_k, the greedy choice of δ^k is

$$\delta^k = \begin{cases} \frac{\delta_L+\delta_R}{2} & \text{for } k=1 \text{ or } J_k \text{ symmetric} \\ \frac{2\delta_L+\delta_R}{3} & \text{for } k \neq 1 \text{ and } \frac{\mathrm{d}J_k}{\mathrm{d}\delta} > 0 \\ \frac{\delta_L+2\delta_R}{3} & \text{for } k \neq 1 \text{ and } \frac{\mathrm{d}J_k}{\mathrm{d}\delta} < 0. \end{cases} \tag{9}$$

The resulting marginal increase ΔA_k in the aggregate performance is

$$\Delta A_k = \begin{cases} \frac{3}{4}\theta(\delta_R - \delta_L)^2 & \text{for } k=1 \\ \frac{1}{8}\theta(\delta_R - \delta_L)^2 & \text{for } k \neq 1 \text{ and } J_k \text{ symmetric} \\ \frac{1}{3}\theta(\delta_R - \delta_L)^2 & \text{otherwise.} \end{cases} \tag{10}$$

A complete proof appears in Appendix A, based on geometric properties of $J_k(\delta)$ and the linear-gap assumptions.

Leveraging the above-mentioned assumptions and the estimated performance function, we propose the *GTTL* algorithm (Algorithm 1). This iterative algorithm selects the optimal hold duration to train on at each step. Initially, under Assumption 4, the optimal training point is the median of the hold-duration range. For subsequent steps, the selection is guided by theorem 1. The transfer process continues until either the aggregate performance area is sufficiently covered or the transfer budget is exhausted. Fig. 5 compares the TTL algorithms in terms of their transfer procedures.

Algorithm 1 starts by initializing with the minimum and maximum hold duration, setting the performance of all hold duration to 0, initializing J and S to 0, and having an empty set for the policies. It continues as long as the covered area is below a threshold or the number of source tasks is less than the budget. For simplicity in notation, we propose substituting the whole area of $(\delta_{\max} - \delta_{\min})J^*$ with A^*. Inside the loop, the

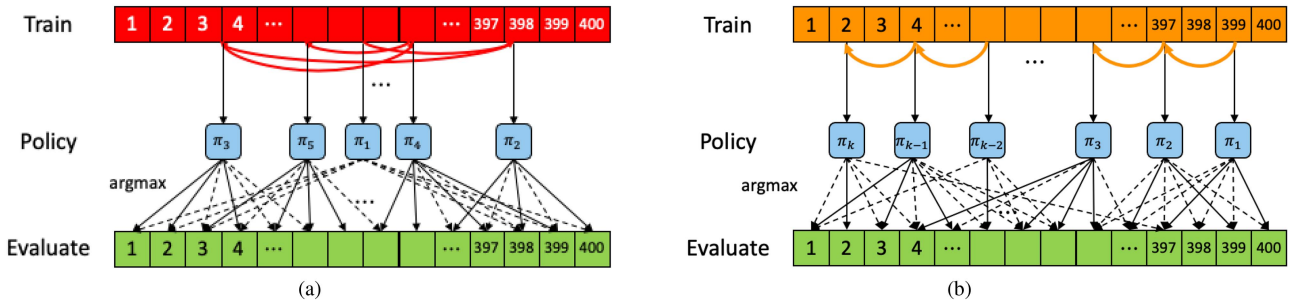


Fig. 5. Illustrative figure of TTL algorithms: Selecting the training task based on the TTL algorithm, evaluating each task based on the trained policies, and taking the best-performing policy for each task. (a) GTTL. (b) CTTL.

Algorithm 2: Find Greedy Transfer Point.

Function FindGreedyTransferPoint($S_k, J_k, \delta_{\min}, \delta_{\max}$)

Cut a range of hold duration $[\delta_{\min}, \delta_{\max}]$ into the segments split by $\delta^k \in S_k$
// Choose δ^k within the segment of $[\delta_L, \delta_R]$ based on the slope of J_k (Theorem 1)
if J_k is symmetric **then**
  $\delta^k \leftarrow \frac{\delta_L+\delta_R}{2}$
else if J_k has positive slope **then**
  $\delta^k \leftarrow \frac{2\delta_L+\delta_R}{3}$
else if J_k has negative slope **then**
  $\delta^k \leftarrow \frac{\delta_L+2\delta_R}{3}$
end if
return δ^k

algorithm chooses a new training task with a hold duration of δ^{k+1} and appends it to its set. It then trains a policy for this hold duration and adds it to the set of policies. After updating the performance with this new policy, the algorithm then calculates the area under this performance curve. Once the loop finishes, the algorithm returns the best performance for each task (J_k) and a set of selected training tasks (S_k). In Algorithm 1, Algorithm 2 assists in identifying the greedy training source task, drawing insights from the shape of the estimated performance function J. This decision-making rule is grounded in theorem 1.

GTTL algorithm (Algorithm 1) exhibits several noteworthy characteristics underpinning its functionality and efficiency. This algorithm is formulated as an anytime algorithm, meaning it can provide a valid solution even if stopped in the middle of the iterations. Beyond mere validity, GTTL algorithm offers performance assurances. At any given step k, GTTL not only provides a valid solution but also ensures a performance that is oriented toward optimization. For example, CTTL might struggle with finer tasks in the initial selection of the source task. This is because the trained policy is inherently skewed to excel in coarser tasks. This means that while other methods, such as CTTL, can also deliver valid results at step k, GTTL is specifically designed to offer a performance closer to optimal at every individual step. This property ensures flexibility and usability under varying operational constraints, allowing continuous solution improvement with each additional source task. This intelligent selection process ensures efficient knowledge transfer and promotes effective learning across different stages of the algorithm's execution.

D. Theoretical Analysis for Optimal TTL

A natural question is how effective these incremental transfer learning strategies are over multiple iterations. One optimality criterion is the best performance achieved within K steps. Analogous to K-means clustering, where the number of clusters affects both granularity and computational cost, the transfer budget K directly influences the quality of the solution. In practice, we are also interested in the minimum number of source tasks, denoted by $K^*(\varepsilon)$, required to achieve a performance within a suboptimality threshold ε. This metric quantifies the algorithm's efficiency in exploring the solution space.

Definition 3. (Cumulative area under the estimated performance function at each iteration): Let A_k denote the cumulative area under the estimated performance function after k iterations. The optimal number of steps $K^*(\varepsilon)$ is defined as the minimum k such that

$$A_{K^*(\varepsilon)} \geq (1-\varepsilon)A^* \tag{11}$$

where A^* represents the maximum possible aggregate performance.

As iterations progress, the cumulative gain A_k approaches A^*, indicating improved performance coverage with each additional source task. Definition 3 essentially conveys about the optimal K (K^*) that is estimated to cover the area of $(1-\varepsilon)A^*$, having remaining area represented by the ratio of ε. Hence, the definition offers insight into how many steps are required to meet the prespecified level of performance as we iterate. Although evaluating the potential coverage for each monotonic segment is critical, deriving a closed-form solution for the optimal policy is challenging due to varying segment shapes.

For a tractable closed-form analysis, we derive a lower bound performance of GTTL by introducing ghost cells at the boundaries, as depicted in Fig. 6. In this lower bound case, we choose $\delta_{\max}$ and $\delta_{\min}$ for the second and third source tasks, respectively, thereby creating a symmetric V-shaped performance profile for all subsegments. We denote the lower bound of the cumulative area after k iterations as $\tilde{A}_k$.

Lemma 1 (Lower bound of GTTL): For all iterations $k = 1, \ldots, K$, the cumulative area under the estimated performance

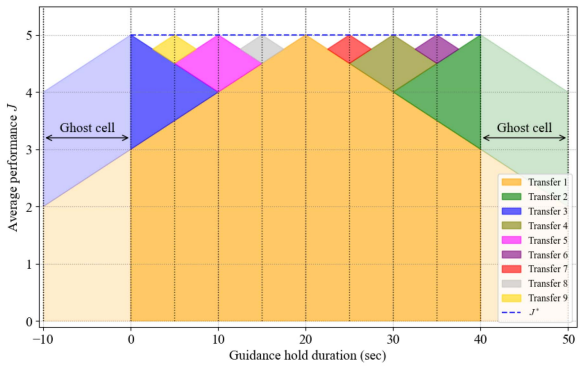


Fig. 6. Illustrative figure for the lower bound of GTTL with the ghost cells at the end of the segments.

function satisfies

$$A_k \geq \tilde{A}_k \quad \forall k = 1, \ldots, K. \tag{12}$$

Proof: Since GTTL always selects the optimal task to maximize the area at each step, the cumulative area A_k is always at least as large as the lower bound area $\tilde{A}_k$, which is computed using suboptimal (ghost cell) choices. □

It follows that if there exists an integer n for which $\tilde{A}_n \geq (1-\varepsilon)A^*$, then $A_n \geq (1-\varepsilon)A^*$, since $A_n \geq \tilde{A}_n$. In fact, at least $\frac{4\varepsilon+1}{4\varepsilon}$ steps are required to cover $(1-\varepsilon)A^*$.

Theorem 2: [The number of source tasks required to cover the area]: If there exists an integer n such that $\tilde{A}_n \geq (1-\varepsilon)A^*$, then $A_n \geq (1-\varepsilon)A^*$. Moreover, at least $\frac{4\varepsilon+1}{4\varepsilon}$ iterations are necessary to achieve this performance threshold.

We prove theorem 2 by leveraging the lower bound cumulative area of GTTL as outlined in Lemma 1, which has a streamlined expression of A_n. The comprehensive proof is provided in Appendix B.

E. Bounded Suboptimality

If the transfer budget is known in advance, a more structured algorithm than GTTL can be designed. This motivates our introduction of CTTL, which selects source tasks uniformly across the hold-duration range. CTTL begins with coarser tasks and progressively transitions to finer ones. For example, with a budget of seven source tasks and a hold-duration range from 1 to 40, training might begin with a hold duration of approximately 37.14, then proceed to 31.43, 25.71, 20, 14.29, 8.57, and finally 2.86, reflecting a diminishing granularity [see Fig. 7(b)].

The advantage of starting with coarser tasks lies in their limited effective horizon, making them easier to solve. Prior work suggests that a coarse-to-fine transfer learning approach is beneficial when adapting a pretrained policy from a coarser to a finer task [42], [43].

Lemma 2 states the optimality of the CTTL algorithm, which selects its subsequent transfer task contingent on the allocated transfer budget K.

Lemma 2: [Optimality of CTTL]: CTTL algorithm establishes the optimality under Assumptions 1–5. Starting from the coarsest task withhold duration $(\delta_{\max} - \frac{\delta_{\max}-\delta_{\min}}{2K})$, CTTL transitions to finer tasks with uniform spacing of $\frac{\delta_{\max}-\delta_{\min}}{K}$. The optimal estimated performance after K iterations is given by

$$A_K^{\mathrm{CTTL}} = \left(1 - \frac{1}{4K}\right)\theta\left(\delta_{\max} - \delta_{\min}\right)^2. \tag{13}$$

The optimality of CTTL can be established via the equality condition of the Cauchy–Schwarz inequality; see Appendix C for details.

In practice, when the optimal number of source tasks $K^*(\varepsilon)$ is known, CTTL tends to outperform GTTL. However, precise knowledge of the transfer budget is often unavailable, making it important to quantify the suboptimality gap between GTTL and the oracle-like CTTL. Theorem 3 provides bounds on the suboptimality of GTTL relative to CTTL for a given transfer budget K.

Theorem 3: [Suboptimality of GTTL]: The suboptimality gap of GTTL relative to CTTL is bounded by

$$\begin{cases} \frac{1}{4K(K-1)}\theta(\delta_{\max}-\delta_{\min})^2 & \text{for } K = 2^i + 1 \\ \frac{1}{2(K-1)^2}\theta(\delta_{\max}-\delta_{\min})^2 & \text{otherwise} \end{cases} \tag{14}$$

where $i \in \mathbb{N}_0$.

The detailed proof is provided in Appendix D, where the suboptimality bounds of GTTL relative to CTTL are derived for the two cases specified previously.

Algorithm 3: Coarse-to-Fine Temporal Transfer Learning (CTTL).

Input: MDP $\mathcal{M}_\delta$, Range of hold duration $[\delta_{\min}, \delta_{\max}]$, Transfer budget K
Output: J_k and S_k
Initialize : $J_0(\delta) = 0\ \forall \delta \in [\delta_{\min}, \delta_{\max}]$, $S_k = \{\}$, $\boldsymbol{\pi} = \{\}$, $k = 0$

while $k \leq K$ **do**
  $\delta^{k+1} = \delta_{\max} - \frac{2k+1}{2K}(\delta_{\max} - \delta_{\min})$
  $S_{k+1} \leftarrow S_k \cup \{\delta^{k+1}\}$
  $\pi_{k+1} \leftarrow \mathrm{Train}(\mathcal{M}(\delta^{k+1}))$
  $\boldsymbol{\pi} \leftarrow \boldsymbol{\pi} \cup \{\pi_{k+1}\}$
  $J_{k+1}(\delta^{k+1}) \leftarrow J^{\pi_{k+1}}(\delta^{k+1})$
  $J_{k+1}(\delta) \leftarrow \max(J_k(\delta), J_{k+1}(\delta^{k+1}) - J_{\delta^{k+1}\to\delta})$
    $\forall \delta \in (\delta_{\min}, \delta_{\max}) \backslash \delta^{k+1}$
  $k \leftarrow k + 1$
end while
return J_k and S_k

V. Simulation Experiments

This section elucidates the simulation experiments conducted to address our primary research questions. The main purpose of our investigation is to explore the potential of human-compatible control serving as an immediate surrogate for AVs and to verify the degree to which such control can optimize traffic performance at a system level. We conducted many experimental trials in various environments to obtain valuable answers to these essential questions.

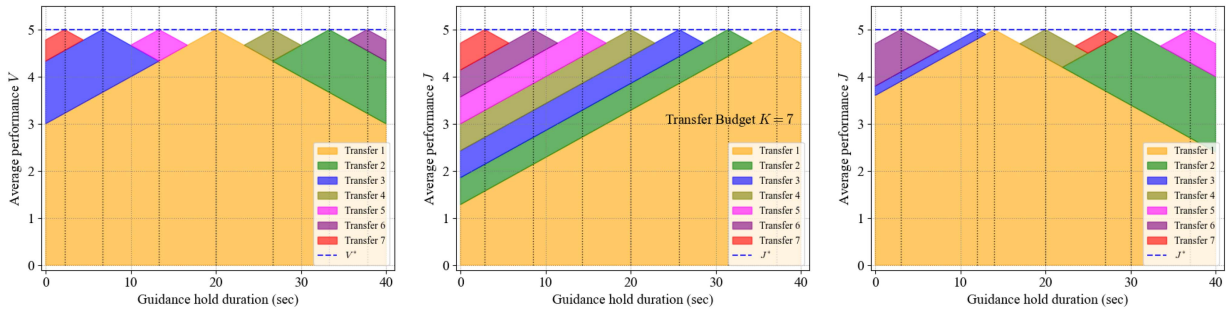


Fig. 7. Illustrative figures for comparing marginal area increase at each iteration by GTTL, CTTL for a given budget K, and RTTL. (a) GTTL. (b) CTTL. (c) RTTL.

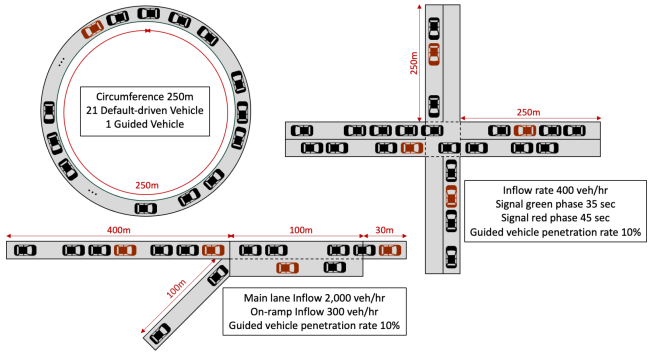


Fig. 8. Modular road networks. Three traffic scenarios for mixed autonomy roadway settings: single-lane ring (top left), highway ramp (bottom), and signalized intersection (top right).

A. Modular Road Networks

In mixed-autonomy roadway settings, we delve into various traffic scenarios as explored in prior works [4], [5], [6], including single-lane ring, highway ramp, and signalized intersection networks, as depicted in Fig. 8. Each scenario has distinct objectives; for instance, the single-lane ring and intersection aim to elevate all vehicles' average velocity, while the highway ramp scenario focuses on increasing the outflow given a constant inflow. The signalized intersection scenario employs a multitask RL strategy, simulating varied penetration rates to accommodate different levels of human-guided vehicle presence, with an evaluation of a penetration rate of 0.1 to assess the RL policy's performance. Default-driven vehicles, which are not equipped with the guidance system, adhere to the intelligent driver model (IDM) for car-following [57].

In the single-lane scenario, the state space is defined by the ego vehicle's speed, the leading vehicle's speed, and the headway. For the highway ramp scenario, the state includes the ego vehicle's speed, relative positions and speeds of the leading and following vehicles in the same lane, and those of the following vehicle on the ramp. In the intersection scenario, the state comprises the ego vehicle's speed, the distance remaining to the intersection, the traffic signal phase, as well as the relative positions and speeds of the leading, following, and adjacent vehicles, the current speed limit, and lane and road identifiers.

The action space varies with the type of guidance employed. For acceleration guidance, a continuous action space ranging from -1 to 1 is utilized, directly influencing the vehicle's acceleration up to a maximum of 2.5 m/s^2. For speed guidance, a discrete action space is defined, with ten actions ranging from 0 to 1, where the chosen action is scaled by the speed limit to determine the vehicle's target speed. The respective reward functions are tailored to the objectives of each scenario. The reward functions for the single-lane ring and highway ramp are the average speed and throughput of the system, respectively. In the signalized intersection scenario, the reward function is defined as the average speed of all vehicles alongside other factors, such as stopping time, abrupt acceleration, and fuel consumption. A thorough examination of these scenarios and reward formulations is provided in Appendix E.

B. Experimental Setup

We utilize the microscopic traffic simulation called Simulation of Urban MObility (SUMO) [58] v.1.16.0 and its accompanying Python API, TRACI, to establish a dynamic link between our algorithmic framework and the SUMO environment. This integration is critical for implementing and testing our traffic management strategies in a controlled, simulated setting. The experiments used the MIT Supercloud with 48 CPU cores [59]. For our numerical experiments, we employed the trust region policy optimization algorithm [60], coupled with a multilayer perceptron neural network architecture featuring two hidden layers, each with 64 units, and employing the $\tanh$ activation function. We trained and tested two different types of guidance: continuous action space for acceleration guidance and discretized action space for speed guidance. We evaluated the system's performance over a range of guidance hold duration $\delta \in [0.1, 40]$. In our simulation experiments, we simplify the analysis by setting $\delta_{\min} = 0$, and we round the calculated δ to the nearest multiple of 10 when selecting the source training task. The detailed experimental setup is explained in Fig. 8 and Appendix E.

C. Baselines

We compare our TTL approaches with several baselines. These baselines represent various strategies for learning and

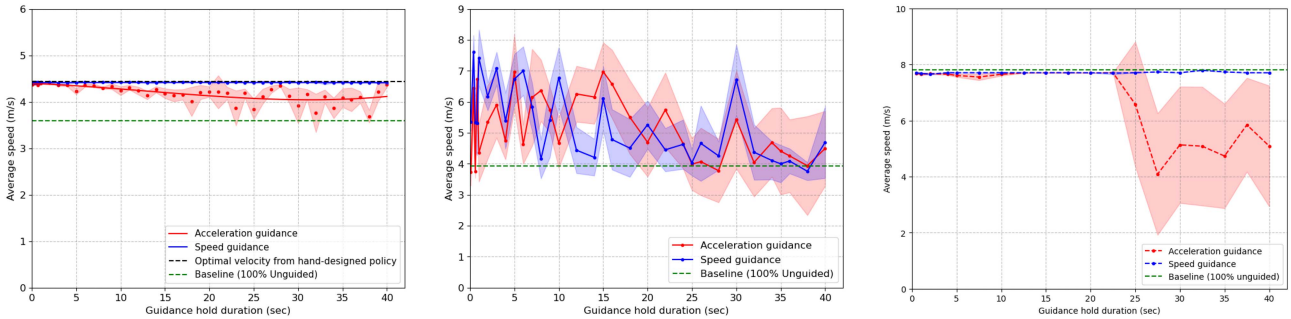


Fig. 9. System performance (average speed of all vehicles) for three traffic scenarios in mixed autonomy roadway settings. Each task with different hold durations is trained exhaustively. (a) Single-lane ring. (b) Highway ramp. (c) Signalized intersection.

TABLE II
COMPARISON OF AVERAGE SPEEDS ACHIEVED BY DIFFERENT TRAINING METHODS ACROSS VARIOUS TRAFFIC SCENARIOS AND GUIDANCE TYPES

Methods	Training complexity	Number of source tasks k	Traffic scenarios: Single-lane ring Accel. guidance	Single-lane ring Speed guidance	Highway Ramp Accel. guidance	Highway Ramp Speed guidance	Signalized Intersection Accel. guidance	Signalized Intersection Speed guidance
Oracle								
Oracle transfer	$n \times n$	n	4.10	4.41	5.48	6.30	7.71	7.71
Exhaustive RL	n	-	3.84	4.29	4.24	4.99	6.86	7.69
Baselines								
100% unguided	0	-	3.80	3.80	3.95	3.95	6.84	6.84
Multitask RL	n†	-	3.94	4.28	4.53	4.42	-	-
TTL (ours)								
CTTL	k	5‡	3.85	4.39	4.47	5.31	**7.71**	**7.71**
	k	10‡	4.02	**4.40**	5.19	5.44	**7.71**	**7.71**
	k	15‡	3.92	4.39	**5.20**	5.56	**7.71**	**7.71**
GTTL	k	5	3.89	4.39	5.19	5.19	**7.71**	**7.71**
	k	10	4.03	**4.40**	5.19	5.54	**7.71**	**7.71**
	k	15	**4.04**	**4.40**	5.19	**6.25**	**7.71**	**7.71**
Ablation								
RTTL	k	5	3.85	4.36	4.53	5.70	7.69	**7.71**
	k	10	3.97	4.39	4.89	5.87	7.70	**7.71**
	k	15	4.01	4.39	5.09	6.03	**7.71**	**7.71**

†: denotes the number of tasks used in multitask RL. It was evaluated after the same number of rollouts of training as other settings.
‡: The performance of the CTTL is assessed after completing the training across a predetermined number of source tasks at budget.

transfer in the context of coarse-grained advisory autonomy tasks.

1) 100% Unguided: In this baseline, all vehicles are unguided, following the IDM car-following models. This scenario represents a completely decentralized system without any RL.

2) Oracle Transfer: The oracle transfer benchmark is an idealized scenario where we train separate models for each possible source task. Once trained, we execute a zero-shot transfer by applying each model to every target task, selecting the most successful model for each.

3) Exhaustive RL: This strategy represents the classic approach to machine learning, where each task is trained individually and evaluated with its corresponding in-distribution trained model. The performance is evaluated by calculating the average performance across all tasks. Fig. 9 illustrates the average speed of all vehicles for three traffic networks trained exhaustively.

4) Multitask RL: The multitask RL framework is designed to simultaneously train a single policy across various tasks [6] and [40], here specifically by varying the guidance hold duration between 1 and 40 s. Each worker in the training process is assigned a specific task and contributes to a shared experience buffer. Upon completion of rollouts by all workers, the policy is updated based on the collective data in the buffer. This approach aims to explore the potential synergies and tradeoffs that arise when a policy is exposed to multiple tasks during the learning process, potentially leading to more robust and generalizable policies.

5) Random Temporal Transfer Learning (RTTL): RTTL chooses tasks for transfer from a pool of temporal tasks at random. This scenario represents a nondeterministic transfer learning strategy and serves as a stochastic comparison point for our deterministic GTTL approach. From all the policies from the source task at each iteration, we select the top-performing one for tasks with varying hold duration [see Fig. 7(c)].

D. TTL Results

Table II and Fig. 11 illustrate the outcomes of TTL algorithms compared to baselines in different traffic environments. Here, the selected performance metric is the average speed of all vehicles evaluated in all tasks. In scenarios such as the highway ramp, while outflow is the primary reward metric, we also present

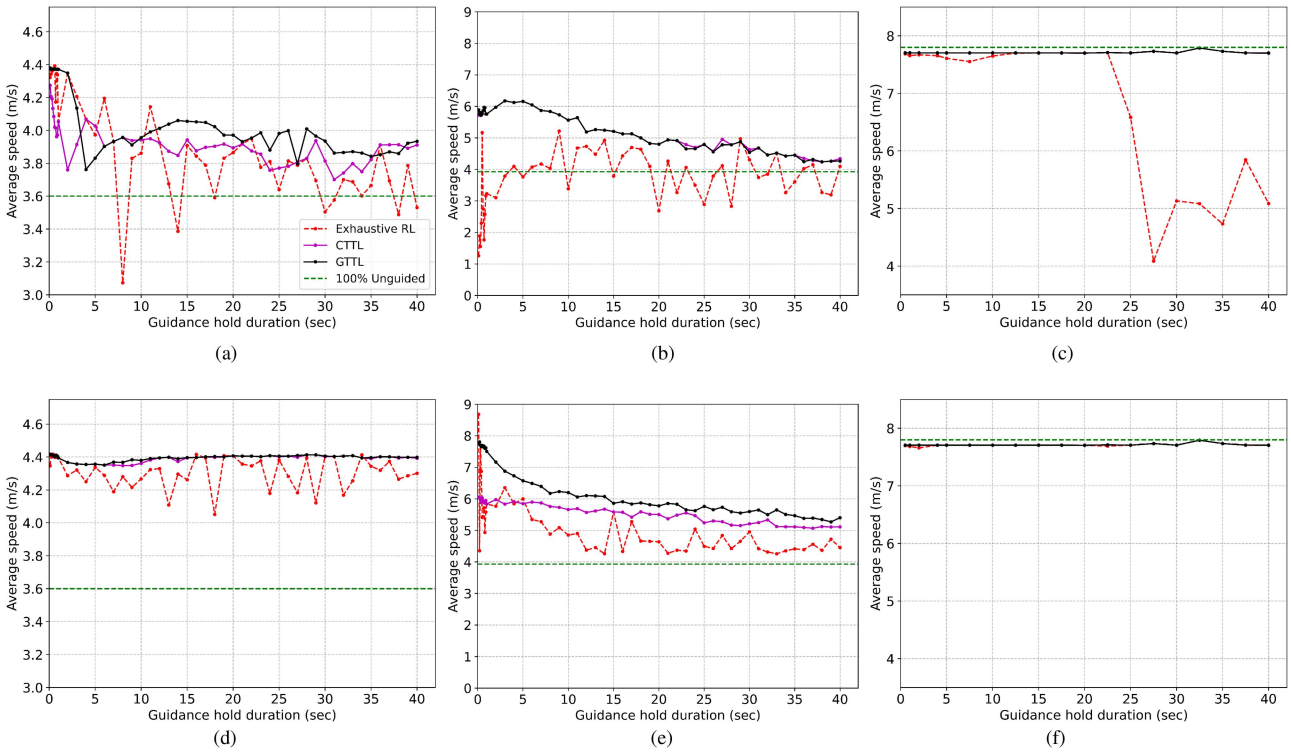


Fig. 10. System performance of TTL algorithms (GTTL and CTTL) compared to the exhaustive RL. The advisory system used either acceleration or speed guidance. (a) Single lane ring with acceleration guidance. (b) Highway ramp with acceleration guidance. (c) Signalized intersection with acceleration guidance. (d) Single lane ring with speed guidance. (e) Highway ramp with speed guidance. (f) Signalized intersection with speed guidance.

the average speed as a metric for comparison. This choice is justified by the high correlation between speed and outflow, providing a consistent measure across different scenarios for a more straightforward comparative analysis. The bolded values represent the highest performance metrics achieved by TTL algorithms, discounting the oracle transfer due to its prohibitive computational demand. The TTL algorithms exhibit exemplary performance in both acceleration and speed guidance categories, markedly outperforming the baselines across diverse traffic conditions. Remarkably, with few source tasks, TTL algorithms approach the near-term performance of the oracle transfer. Some scenarios require a small number of source tasks to achieve the near-term performance of oracle transfer, while others demand more extensive iterations. Specifically, in the signalized intersection scenario, all transfer learning methods yield the top performance when paired with speed guidance. This highlights the effectiveness of TTL in optimizing traffic management tasks, particularly when combined with speed guidance. Fig. 10 shows the system-level performance of each task after the TTL methods is applied, compared to the exhaustive RL.

The results presented in Fig. 11 offer an insightful comparison of several training methodologies in the context of coarse-grained advisory autonomy tasks in different traffic scenarios. The performance metrics present in Fig. 11 indicate the average of evaluated performance across the full range of the coarse-grained advisory, and each task is evaluated with the average speed of all vehicles, with higher values denoting better performance. It provides a gauge for the generalizability of source tasks selected from different methods across various target tasks.

Single-lane ring: Fig. 9(a) compares the system performance of acceleration and speed guidance in the single-lane ring road network when trained from scratch. When analyzing the results, both guidance types demonstrate excellent overall performance as the guidance hold duration increases, with an average speed increase of approximately 22.22% for all vehicles in the system. However, it is worth noting that the acceleration guidance results were slightly lower than speed guidance. First, in a single-lane ring environment [see Fig. 11(a)], the average speed of GTTL starts higher than both RTTL and CTTL in the first iteration of selecting the source task. Despite a slight decrease in the early stages, the speed improves consistently over the iterations and stays competitive against the other methods. The performance of GTTL shows that it learns quickly in the initial stages and then continues to optimize its performance in subsequent steps, indicating an effective transfer of knowledge. It is also worth noting that while no strategy surpasses oracle transfer's average speed of 4.10 m/s, GTTL gets relatively close, reaching final average speeds of approximately 4.04 m/s. While the trends for RTTL are upward as the number of source tasks gets larger, it does not exceed the performance demonstrated by GTTL. Multitask RL, although slightly surpassing the baseline, falls short when compared to our GTTL method.

Furthermore, a clear distinction is observed when comparing the number of source tasks required to achieve a given

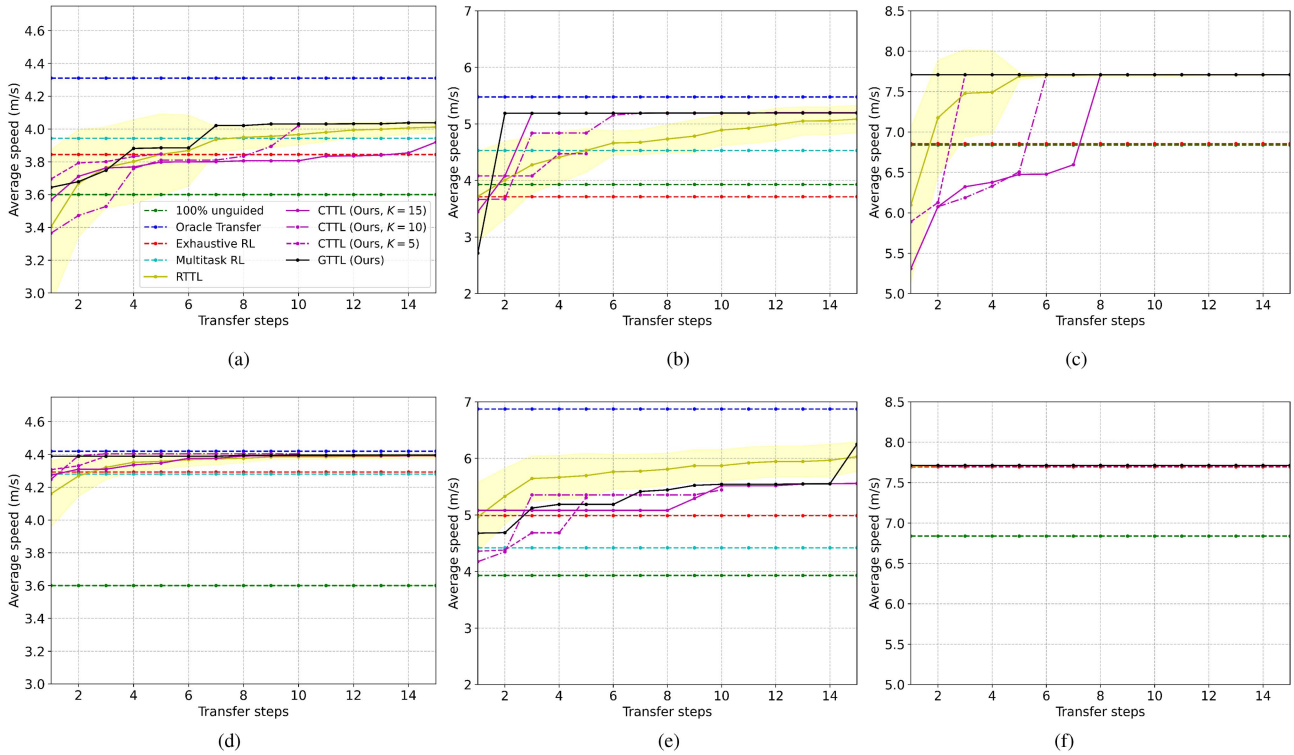


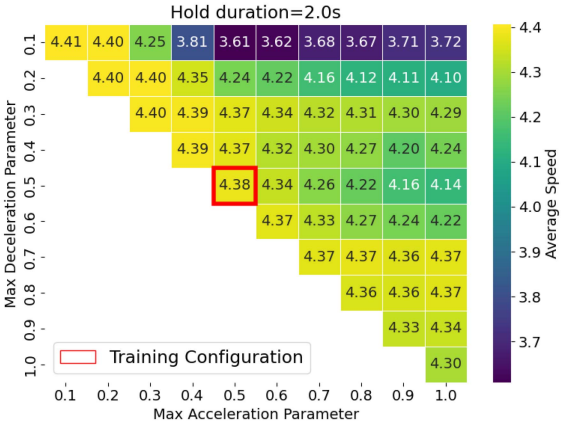
Fig. 11. System performance comparison of TTL with various baselines in three different traffic scenarios and two different guidance types: unguided vehicles (representing a completely decentralized system), oracle transfer (showcasing zero-shot transfer capabilities), exhaustive RL (traditional separate model per task approach), multitask RL (incorporating multiple tasks into the learning process), and transfer learning strategies such as greedy (choosing the one-step greedy source training task) and coarse-to-fine (transferring from coarser to finer tasks progressively). (a) Single lane ring with acceleration guidance. (b) Highway ramp with acceleration guidance. (c) Signalized intersection with acceleration guidance. (d) Single lane ring with speed guidance. (e) Highway ramp with speed guidance. (f) Signalized intersection with speed guidance.

performance level across methods. To surpass baselines with exhaustive RL, RTTL necessitates approximately ten steps, whereas GTTL achieves this in merely seven steps, highlighting its efficiency. These findings strongly advocate the effectiveness of GTTL in such driving scenarios, reinforcing its potential suitability for real-world applications in achieving coarse-grained advisory in mixed autonomy. Upon examining speed guidance results [see Fig. 11(d)], we observe that performance levels are already near-optimal even before applying transfer learning algorithms. This observation highlights the intrinsic effectiveness of speed guidance, making the added benefits derived from implementing TTL algorithms less distinguishable in this specific scenario.

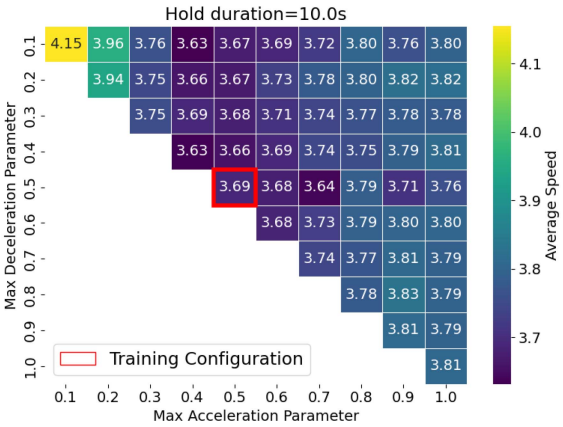
Highway ramp: Following the single-lane ring road scenario, we analyze the results from a highway ramp scenario, where the complexity of the traffic situations and interactions are significantly elevated. Fig. 9(b) displays the jagged performance of training exhaustively in the highway ramp road network, which could indicate the difficulty of traffic coordination around the ramp and brittleness of RL training. However, the overall trend suggests that the average speed of all vehicles decreases as the guidance hold duration increases. In both scenarios, the multitask RL approach—trained with access to all target tasks—demonstrates a slight improvement over the unguided baseline but still falls significantly short of the performance offered by the TTL algorithms.

With the acceleration guidance [see Fig. 11(b)], oracle transfer, considered as upper bound performance, consistently achieved 5.48 m/s for speed guidance, while the average speed in the unguided case is maintained around 3.95 m/s. CTTL progressively improved the average speed from 4.47 m/s within the budget of 5–5.20 m/s over 15 source tasks, obtaining the highest performance. GTTL started at 5.19 m/s after the first five steps, which is the highest among other methods, and eventually optimized its performance to 5.19 m/s across the 15 steps. Switching to the speed guidance scenario [see Fig. 11(e)], all methods indicated an enhancement compared to the acceleration guidance scenario. The RTTL method started at 5.70 m/s and reached a higher peak speed of 6.03 m/s. Furthermore, the CTTL method increased the average speed from 5.31 to 5.56 m/s over the 15 steps. GTTL exhibited robustness, initiating at an average speed of 5.19 m/s and advancing to 6.25 m/s across the steps. This result where RTTL outperforms GTTL and CTTL highlights the unpredictable aspects of RL training, indicating that despite GTTL's strategic framework, random task selection by RTTL can, at times, yield comparable or superior results.

Signalized intersection: Analyzing the signalized intersection scenarios with acceleration and speed guidance reveals some notable trends. When trained exhaustively, the system performance of the signalized intersection remains steady [see Fig. 9(c)]. During the training of the multitask RL policy, achieving a robust



(a)



(b)

Fig. 12. *Robustness to different IDM parameters:* Heatmaps of average speed (m/s) under different combinations of max-acceleration (x-axis) and max-deceleration (y-axis) parameters. Warmer colors represent higher speeds. Both short (2 s) and longer (10 s) hold durations achieve stable performance across diverse driver profiles. (a) Guidance hold duration 2 s. (b) Guidance hold duration 10 s.

policy that could generalize across all ranges of hold duration tasks turned out to be challenging. Once the policy converged, it unfortunately led to an increase in collisions at intersections.

For acceleration guidance [see Fig. 11(c)], the unguided scenario and exhaustive RL resulted in average speeds of 6.84 and 6.86 m/s, respectively, while oracle transfer reached 7.71 m/s. TTL methods, particularly CTTL and GTTL, improved significantly, up to the performance of Oracle Transfer. Speed guidance scenario [see Fig. 11(f)] benefits from the transfer learning procedures. Both CTTL and GTTL mirrored Oracle Transfer performance, achieving average speeds of around 7.71 m/s, almost close to the optimal performance. We observe that at signalized intersections with speed guidance, even a single source task allows transfer learning methods to match the performance of the oracle. This may be due to the more predictable nature of traffic dynamics at intersections, making them amenable to successful transfer with minimal learning. Moreover, the selection of the K parameter for CTTL should be strategically determined by

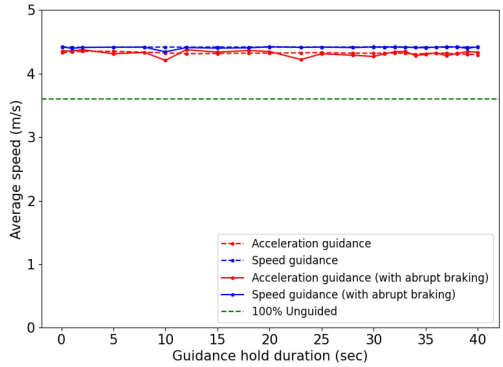


Fig. 13. *Resilience from abrupt braking:* Comparison of our acceleration- and speed-guidance strategies trained with TTL and "abrupt braking" in a single-lane ring scenario, along with a 100% unguided baseline (green dashed line). Coarse-grained guidance maintains a higher average speed even under abrupt driver actions.

balancing the computational budget and the complexity of the task to optimize the algorithm's performance.

E. Sensitivity Analysis

While the IDM is a commonly used car-following model in mixed-autonomy research; it does not capture every nuance of real-world human driving. To evaluate how our learned policies perform under more diverse and less "ideal" driver behaviors, we conducted two complementary sensitivity analyses: 1) randomizing key IDM parameters, and 2) inducing abrupt braking events in a subset of human-driven vehicles.

Randomized IDM parameters: We first examine how variations in the maximum acceleration and deceleration parameters affect overall system performance. Fig. 12 demonstrates that the learned policies using TTL generalize well beyond a single set of IDM parameters to a broad range of aggressive and conservative driver profiles, both in short and long guidance hold duration.

Abrupt braking events: In addition to IDM parameter variations, we also tested the resilience of our policy against unexpected driver actions by injecting "abrupt braking" behaviors. Specifically, a few human-driven vehicles were programmed to perform sudden stops with low probability, disrupting the smooth car-following patterns. Fig. 13 shows how our acceleration- and speed-guidance policies sustain high average speed in the single-lane ring scenario despite these abrupt disruptions, underscoring the policy's ability to handle erratic driver maneuvers.

Overall, these experiments confirm that our learned guidance strategies remain effective under both parameter variations and abrupt disruptions, underscoring their robustness and potential for real-world mixed-autonomy applications.

VI. Conclusion

This article presents TTL algorithms for coarse-grained advisory, addressing the intrinsic complexity and brittleness of RL algorithms. Through empirical analysis across three traffic scenarios, we evaluate the performance of the advisory system

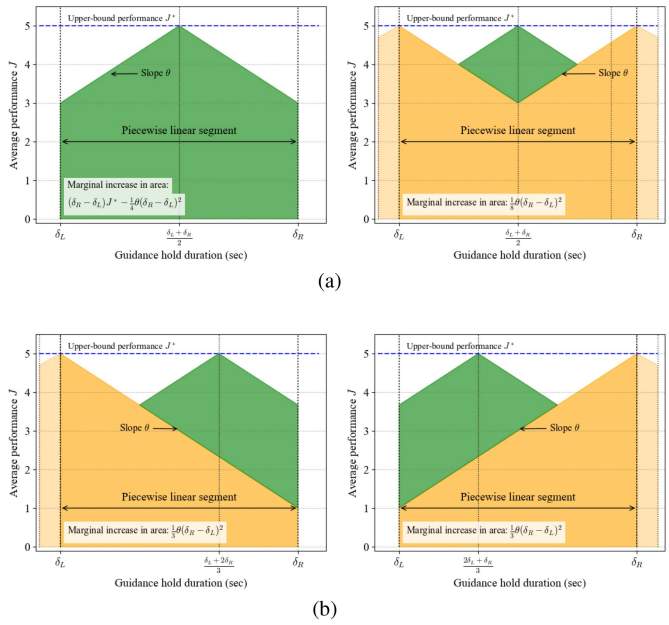


Fig. 14. Decision strategies and corresponding marginal performance increases for symmetric and asymmetric $J_k(\delta)$. (a) When $J_k(\delta)$ is symmetric about the center, the optimal δ_k is the midpoint, maximizing the area under J_k. (b) For asymmetric $J_k(\delta)$, the trisection points yield the optimum depending on the local slope. The green region indicates the marginal gain from the current choice, contrasting with prior coverage (orange). (a) $J_k(\delta)$ is symmetric. (b) $J_k(\delta)$ is asymmetric.

through either acceleration or speed and see how much performance is near-term as instantaneous control. Significant findings show that TTL outperforms baselines in diverse traffic conditions, indicating its potential to improve traffic management in mixed-autonomy environments and reduce computational cost for training RL policies. TTL algorithm is also generic and applicable to different contextual MDP tasks and can find meaningful cross-domain applications especially in industrial automation and robotics. Relaxing our initial assumptions, such as the constant estimation of training performance, could enhance our algorithm's applicability and effectiveness. Future work will explore more complex scenarios and relax the theoretical assumptions to bridge the gap toward real-world applicability. We also underscore the potential into computationally efficient continual learning and fine-tuning strategies for transfer learning, which could offer a viable way for improved traffic management system.

Appendix A
Proof for Theorem 1

Proof: Fig. 14 illustrates the one-step decision for selecting the next hold duration δ_k to maximize the marginal performance gain within a segment $[\delta_L, \delta_R]$. Let $\theta_L, \theta_R > 0$ denote the local (piecewise-linear) slopes near the boundaries. We consider three cases.

Case 1: Symmetric $J_k(\delta)$: Define the objective as the sum of the two trapezoidal areas created by a split at $\delta' \in [\delta_L, \delta_R]$

$$\max_{\delta'} \text{Obj}(\delta') = \frac{1}{2}(\delta' - \delta_L)[2J^* - \theta_L(\delta' - \delta_L)] + \frac{1}{2}(\delta_R - \delta')[2J^* - \theta_R(\delta_R - \delta')]$$
$$= (\delta_R - \delta_L)J^* - \frac{\theta_L}{2}(\delta' - \delta_L)^2 - \frac{\theta_R}{2}(\delta_R - \delta')^2.$$

Setting $\frac{\mathrm{d}}{\mathrm{d}\delta'}\text{Obj}(\delta') = 0$ gives

$$-\theta_L(\delta' - \delta_L) + \theta_R(\delta_R - \delta') = 0 \quad \Rightarrow \quad \delta' = \frac{\theta_L\delta_L + \theta_R\delta_R}{\theta_L + \theta_R}.$$

Under Assumption 4 ($\theta_L = \theta_R = \theta$), the optimizer is the midpoint $\delta' = \frac{\delta_L+\delta_R}{2}$. The resulting marginal gain is $\Delta A_k = \frac{1}{8}\theta(\delta_R - \delta_L)^2$.

Case 2: Positive slope: For a positively sloped $J_k(\delta)$ across the segment, the objective reduces to

$$\text{Obj}(\delta') = \frac{1}{2}\theta(\delta_R + 3\delta' - 2\delta_L)(\delta_R - \delta').$$

Differentiating and setting to zero yields $\delta' = \frac{2\delta_L+\delta_R}{3}$, and the marginal area increase is $\Delta A_k = \frac{1}{3}\theta(\delta_R - \delta_L)^2$.

Case 3: Negative slope: By symmetry, for a negative slope the maximizer is $\delta' = \frac{\delta_L+2\delta_R}{3}$, with the same marginal gain $\Delta A_k = \frac{1}{3}\theta(\delta_R - \delta_L)^2$.

These three cases establish the one-step greedy choice. □

Appendix B
Proof for Theorem 2

Proof: For the initial step

$$\tilde{A}_1 = (\delta_{\max} - \delta_{\min})J^* - \frac{1}{4}\theta(\delta_{\max} - \delta_{\min})^2.$$

From the geometry in Fig. 6, the next few odd steps satisfy

$$\tilde{A}_3 = (\delta_{\max} - \delta_{\min})J^* - \frac{1}{8}\theta(\delta_{\max} - \delta_{\min})^2$$
$$\tilde{A}_5 = (\delta_{\max} - \delta_{\min})J^* - \frac{1}{16}\theta(\delta_{\max} - \delta_{\min})^2$$
$$\tilde{A}_9 = (\delta_{\max} - \delta_{\min})J^* - \frac{1}{32}\theta(\delta_{\max} - \delta_{\min})^2, \ \ldots$$

In general, for $k = 2^i + 1$ with $i \in \mathbb{N}$ and using Assumption 5, i.e., $J^* = \theta(\delta_{\max} - \delta_{\min})$

$$\tilde{A}_{2^i+1} = (\delta_{\max} - \delta_{\min})J^* - \frac{1}{2^{i+2}}\theta(\delta_{\max} - \delta_{\min})^2$$
$$= \left(1 - \frac{1}{2^{i+2}}\right)\theta(\delta_{\max} - \delta_{\min})^2.$$

To cover at least a fraction $(1 - \varepsilon)$ of the full area

$$\tilde{A}_{2^i+1} \geq (1 - \varepsilon)\,\theta(\delta_{\max} - \delta_{\min})^2 \iff \varepsilon \geq \frac{1}{2^{i+2}}.$$

Hence

$$2^i + 1 \ \geq \ \frac{1}{4\varepsilon} + 1 \ = \ \frac{4\varepsilon + 1}{4\varepsilon}$$

so at least $\frac{4\varepsilon+1}{4\varepsilon}$ steps are required. □

Appendix C
Proof for Theorem 4

Proof: Consider a transfer budget of K and partition $[\delta_{\min}, \delta_{\max}]$ into $K+1$ subsegments with lengths $l_1, \ldots, l_{K+1}$

(so $\sum_{k=1}^{K+1} l_k = \delta_{\max} - \delta_{\min}$). The remaining area below J^* after the CTTL sequence equals the sum of triangle areas within each subsegment

$$\tfrac{1}{2}\theta l_1^2, \quad \tfrac{1}{4}\theta l_k^2 \ (k = 2, \dots, K), \quad \tfrac{1}{2}\theta l_{K+1}^2.$$

Thus, we minimize

$$\min_{l_k \geq 0} \quad \frac{1}{2}\theta l_1^2 + \frac{1}{4}\theta l_2^2 + \cdots + \frac{1}{4}\theta l_K^2 + \frac{1}{2}\theta l_{K+1}^2$$
$$\text{s.t.} \quad \sum_{k=1}^{K+1} l_k = \delta_{\max} - \delta_{\min}.$$

Let $\mathbf{u} = (l_1/\sqrt{2}, l_2/2, \dots, l_K/2, l_{K+1}/\sqrt{2})$ and $\mathbf{v} = (\sqrt{2}, 2, \dots, 2, \sqrt{2})$. Then, $\langle \mathbf{u}, \mathbf{v} \rangle = \sum_{k=1}^{K+1} l_k = \delta_{\max} - \delta_{\min}$ and $\|\mathbf{v}\|^2 = 4K$. By Cauchy–Schwarz

$$\theta \|\mathbf{u}\|^2 \ \geq \ \theta \frac{\langle \mathbf{u}, \mathbf{v} \rangle^2}{\|\mathbf{v}\|^2} = \frac{\theta}{4K} (\delta_{\max} - \delta_{\min})^2.$$

Equality holds when $\mathbf{u} = \lambda \mathbf{v}$, i.e., $2l_1 = l_2 = \cdots = l_K = 2l_{K+1}$. Using the sum constraint

$$l_1 = l_{K+1} = \frac{\delta_{\max} - \delta_{\min}}{2K}, \qquad l_2 = \cdots = l_K = \frac{\delta_{\max} - \delta_{\min}}{K}.$$

Therefore, the optimal objective value is

$$A_K^{\text{CTTL}} = \frac{\theta}{4K} (\delta_{\max} - \delta_{\min})^2$$

which proves the optimality of the CTTL allocation. □

Appendix D
Proof for Theorem 3

Proof: We consider the cases $K = 2^i{+}1$ and $2^{i-1}{+}1 < K < 2^i{+}1$ with $i \in \mathbb{N}$.

Case 1: $K = 2^i{+}1$: From Appendix B

$$A_K^{\text{CTTL}} = \Big(1 - \frac{1}{4K}\Big)\theta(\delta_{\max} - \delta_{\min})^2$$
$$A_K^{\text{GTTL}} \geq \tilde{A}_K^{\text{GTTL}} = \Big(1 - \frac{1}{4(K-1)}\Big)\theta(\delta_{\max} - \delta_{\min})^2.$$

Hence

$$A_K^{\text{CTTL}} - A_K^{\text{GTTL}} \leq \frac{1}{4K(K-1)}\,\theta(\delta_{\max} - \delta_{\min})^2.$$

Case 2: $2^{i-1}{+}1 < K < 2^i{+}1$: Using $\tilde{A}^{\text{GTTL}}_{2^i+1} = (1 - \frac{1}{2^{i+2}})\theta(\cdot)^2$ and $\tilde{A}^{\text{GTTL}}_{2^{i-1}+1} = (1 - \frac{1}{2^{i+1}})\theta(\cdot)^2$, the uniform increments over this range are

$$\tilde{A}_{K+1}^{\text{GTTL}} - \tilde{A}_K^{\text{GTTL}} = \frac{1}{2^{2i+1}}\,\theta(\delta_{\max} - \delta_{\min})^2.$$

Therefore

$$\tilde{A}_K^{\text{GTTL}} = \tilde{A}^{\text{GTTL}}_{2^{i-1}+1} + \frac{K - (2^{i-1}+1)}{2^{2i+1}}\theta(\delta_{\max} - \delta_{\min})^2$$
$$= \Big(1 + \frac{K - 3\cdot 2^{i-1} - 1}{2^{2i+1}}\Big)\theta(\delta_{\max} - \delta_{\min})^2.$$

Then

$$A_K^{\text{CTTL}} - A_K^{\text{GTTL}} \leq A_K^{\text{CTTL}} - \tilde{A}_K^{\text{GTTL}}$$
$$\leq \frac{1}{2^{2i+1}}\,\theta(\delta_{\max} - \delta_{\min})^2$$
$$\leq \frac{1}{2(K-1)^2}\,\theta(\delta_{\max} - \delta_{\min})^2$$

which yields the claimed bound. □

Appendix E
Experimental Details for Modular Road Network

In mixed autonomy roadway settings, we investigate the traffic scenarios covered in the previous works [4], [5], [6], including the following road networks: single-lane ring, highway ramp, and signalized intersection.

a) Single-lane ring: Inspired by Sugiyama [61], the circular ring has a circumference of 250 m. The single-lane ring environment aims to increase the average velocity of all vehicles in the road network.

The reward function is the average speed of all vehicles

$$r(s, a) = \frac{1}{n} \sum_{\forall i} v_i(s, a). \tag{15}$$

b) Highway ramp: The objective in the highway ramp environment was to increase the outflow given the same inflow. The reward function in the highway ramp scenario focuses on traffic flow efficiency. It is defined as the number of vehicles exiting the system. Specifically, we compare the average speed of all vehicles in the system as a performance measure.

c) Signalized intersection: We have designed a single-lane, four-way signalized intersection regulated by a static traffic signal phase. A multitasking training approach is employed to train this intersection. Specifically, we use a multitask RL strategy, considering various penetration rates to simulate different levels of human-guided vehicle presence. Nonetheless, when evaluating the effectiveness of this strategy, we focus on scenarios with a 0.1 penetration rate. This allows us to assess the performance of the trained RL policy in conditions where only 10% of the vehicles are controlled by the RL policy, and the remaining 90% operate under human driving model.

For the signalized intersection, the reward function incorporates multiple components to optimize various aspects of traffic flow

$$r(s, a) = \frac{1}{n} \sum_{\forall i} v_i(s, a) - \alpha_1 \cdot P_{\text{stop}} - \alpha_2 \cdot A_{\text{accel}} - \alpha_3 \cdot F_{\text{consumption}} \tag{16}$$

where:

1) P_{stop} is a penalty for vehicles stopped or moving slower than a set threshold (typically 1 m/s), aimed at reducing delays;
2) A_{accel} penalizes abrupt acceleration or deceleration to encourage smoother driving;
3) $F_{\text{consumption}}$ penalizes high fuel consumption, promoting ecofriendly driving practices.

TABLE III
EXPERIMENTAL PARAMETERS FOR RL, TTL, SIMULATIONS, CAR FOLLOWING MODELS, AND SCENARIOS

Type	Experiment parameters	Value
RL	Training epochs	1000
	Discount factor	0.999
	Test epochs	50
	Number of discrete action space	10
Simulation	Simulation step	0.1 s/step
	Warmup steps	500 s
	Timestep horizon	1000 s
Car following model	Model	IDM [57]
	Maximum acceleration	1 m/s^2
	Comfortable deceleration	1.5 m/s^2
	Desired velocity	30 m/s
	Minimum spacing	2 m
	Desired time headway	1 s
	Exponent	4
Ring	Circumference	250 m
	Number of controlled vehicles	1
	Total number of vehicles	22
	Speed limit	10 m/s
Highway ramp	Mainlane inflow rate	2000 veh/hr
	On-ramp inflow rate	300 veh/hr
	Guided vehicle penetration rate	0.1
	Speed limit	30 m/s
Signalized intersection	Inflow rate	400 veh/hr
	Signal phase time (green, red)	35, 45 s
	Guided vehicle penetration rate	0.1
	Speed limit	14 m/s
	Weight for stop penalty	35
	Weight for acceleration	1
	Weight for fuel consumption	1
TTL	Transfer budget	15

The weights $\alpha_1, \alpha_2, \alpha_3$ are hyperparameters that fine-tune the significance of each penalty within the reward function, allowing for a balanced consideration of speed, safety, and efficiency based on the specific goals of each scenario.

Similar to the highway ramp scenario, we compare the average speed of all vehicles in the system as a performance measure.

Table III provides the detailed experimental setup for RL, microscopic traffic simulation, car-following models, and the traffic scenarios used in the experiments.

ACKNOWLEDGMENT

The authors acknowledge the MIT SuperCloud and Lincoln Laboratory Supercomputing Center for providing HPC resources that have contributed to the research results reported within this article. The authors would like to thank Katie Driggs–Campbell for the insightful discussion about advisory autonomy.

REFERENCES

[1] C. Wu, A. Kreidieh, E. Vinitsky, and A. M. Bayen, "Emergent behaviors in mixed-autonomy traffic," in *Proc. Conf. Robot Learn*. PMLR, 2017, pp. 398–407.

[2] R. E. Stern et al., "Dissipation of stop-and-go waves via control of autonomous vehicles: Field experiments," *Transp. Res. part C: Emerg. Technol.*, vol. 89, pp. 205–221, 2018.

[3] M. Sridhar and C. Wu, "Piecewise constant policies for human-compatible congestion mitigation," in *Proc. IEEE Int. Intell. Transp. Syst. Conf.*, IEEE, 2021, pp. 2499–2505.

[4] Z. Yan and C. Wu, "Reinforcement learning for mixed autonomy intersections," in *Proc. IEEE Int. Intell. Transp. Syst. Conf.*, IEEE, 2021, pp. 2089–2094.

[5] C. Wu, A. R. Kreidieh, K. Parvate, E. Vinitsky, and A. M. Bayen, "Flow: A modular learning framework for mixed autonomy traffic," *IEEE Trans. Robot.*, vol. 38, no. 2, pp. 1270–1286, Apr. 2022.

[6] Z. Yan, A. R. Kreidieh, E. Vinitsky, A. M. Bayen, and C. Wu, "Unified automatic control of vehicular systems with reinforcement learning," *IEEE Trans. Automat. Sci. Eng.*, vol. 20, no. 2, pp. 789–804, Apr. 2023.

[7] J. Kim, J.-H. Cho, and C. Wu, "Reinforcement learning for robust advisories under driving compliance errors," *IEEE Trans. Intell. Transp. Syst.*, vol. 26, no. 6, pp. 7780–7791, Jun. 2025.

[8] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *J. Mach. Learn. Res.*, vol. 10, pp. 1633–1685, Dec. 2009.

[9] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[10] A. R. Kreidieh, C. Wu, and A. M. Bayen, "Dissipating stop-and-go waves in closed and open networks via deep reinforcement learning," in *Proc. 21st Int. Conf. Intell. Transp. Syst*. IEEE, 2018, pp. 1475–1480.

[11] K. Jang et al., "Simulation to scaled city: Zero-shot policy transfer for traffic control via autonomous vehicles," in *Proc. 10th ACM/IEEE Int. Conf. cyber- Phys. Syst.*, 2019, pp. 291–300.

[12] E. Vinitsky, K. Parvate, A. Kreidieh, C. Wu, and A. Bayen, "Lagrangian control through deep-RL: Applications to bottleneck decongestion," in *Proc. 21st Int. Conf. Intell. Transp. Syst.*. IEEE, 2018, pp. 759–765.

[13] R. Bishop, "Intelligent vehicle applications worldwide," *IEEE Intell. Syst. Appl.*, vol. 15, no. 1, pp. 78–81, Jan./Feb. 2000.

[14] K. Katsaros, R. Kernchen, M. Dianati, and D. Rieck, "Performance study of a green light optimized speed advisory (GLOSA) application using an integrated cooperative ITS simulation platform," in *Proc. 7th Int. Wireless Commun. Mobile Comput. Conf.* IEEE, 2011, pp. 918–923.

[15] X. Xiang, K. Zhou, W.-B. Zhang, W. Qin, and Q. Mao, "A closed-loop speed advisory model with driver's behavior adaptability for eco-driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 6, pp. 3313–3324, Dec. 2015.

[16] A. Hasan, N. Chakraborty, H. Chen, J.-H. Cho, C. Wu, and K. Driggs-Campbell, "PeRP: Personalized residual policies for congestion mitigation through co-operative advisory systems," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, 2023, pp. 5078–5085.

[17] A. Hasan, N. Chakraborty, H. Chen, J.-H. Cho, C. Wu, and K. Driggs-Campbell, "Cooperative advisory residual policies for congestion mitigation," *J. Auton. Transp. Syst.*, vol. 2, no. 2, pp. 1–31, 2024.

[18] B. Mok et al., "Emergency, automation off: Unstructured transition timing for distracted drivers of automated vehicles," in *Proc. IEEE 18th Int. Conf. Intell. Transp. Syst*. IEEE, 2015, pp. 2458–2464.

[19] S. Li, R. Dong, and C. Wu, "Integrated analysis of coarse-grained guidance for traffic flow stability," *IEEE Trans. Control Netw. Syst.*, vol. 11, no. 3, pp. 1382–1394, Sep. 2024.

[20] R. S. Sutton, D. Precup, and S. Singh, "Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning," *Artif. Intell.*, vol. 112, no. 1–2, pp. 181–211, 1999.

[21] A. Lakshminarayanan, S. Sharma, and B. Ravindran, "Dynamic action repetition for deep reinforcement learning," *Proc. AAAI Conf. Artif. Intell.*, vol. 31, no. 1, pp. 2133–2139, 2017.

[22] S. Sharma, A. S. Lakshminarayanan, and B. Ravindran, "Learning to repeat: Fine grained action repetition for deep reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2017, pp. 1–24.

[23] A. Biedenkapp, R. Rajan, F. Hutter, and M. Lindauer, "Temporl: Learning when to act," in *Proc. Int. Conf. Mach. Learn*. PMLR, 2021, pp. 914–924.

[24] A. M. Metelli, F. Mazzolini, L. Bisi, L. Sabbioni, and M. Restelli, "Control frequency adaptation via action persistence in batch reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*. PMLR, 2020, pp. 6862–6873.

[25] J. Lee, B.-J. Lee, and K.-E. Kim, "Reinforcement learning for control with multiple frequencies," *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 3254–3264, 2020.

[26] L. Yang, S. Hanneke, and J. Carbonell, "A theory of transfer learning with applications to active learning," *Mach. Learn.*, vol. 90, no. 2, pp. 161–189, 2013.

[27] M. K. Helwa and A. P. Schoellig, "Multi-robot transfer learning: A dynamical system perspective," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*. IEEE, 2017, pp. 4702–4708.

[28] W. M. Kouw and M. Loog, "An introduction to domain adaptation and transfer learning," 2018, *arXiv:1812.11806*.

[29] N. Tripuraneni, M. Jordan, and C. Jin, "On the theory of transfer learning: The importance of task diversity," *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 7852–7862, 2020.

[30] J. Guan and Z. Lu, "Task relatedness-based generalization bounds for meta learning," in *Proc. Int. Conf. Learn. Representations*, 2022, pp. 1–19.

[31] I. Higgins et al., "Darla: Improving zero-shot transfer in reinforcement learning," in *Proc.Int. Conf. Mach. Learn.* PMLR, 2017, pp. 1480–1490.

[32] A. A. Rusu, M. Večerí, T. Rothörl, N. Heess, R. Pascanu, and R. Hadsell, "Sim-to-real robot learning from pixels with progressive nets," in *Proc. 1st Annu. Conf. Robot Learn.*, PMLR, 2017, pp. 262–270.

[33] A. R. Kreidieh, G. Berseth, B. Trabucco, S. Parajuli, S. Levine, and A. M. Bayen, "Inter-level cooperation in hierarchical reinforcement learning," 2019, *arXiv:1912.02368*.

[34] C. K. Man, M. Quddus, and A. Theofilatos, "Transfer learning for spatio-temporal transferability of real-time crash prediction models," *Accident Anal. Prevention*, vol. 165, 2022, Art. no. 106511.

[35] R. Oruche, L. Egede, T. Baker, and F. O'Donncha, "Transfer learning to improve streamflow forecasts in data sparse regions," 2021, *arXiv:2112.03088*.

[36] Y. Jang, H. Lee, S. J. Hwang, and J. Shin, "Learning what and where to transfer," in *Proc. Int. Conf. Mach. Learn.*. PMLR, 2019, pp. 3030–3039.

[37] J. Sinapov, S. Narvekar, M. Leonetti, and P. Stone, "Learning inter-task transferability in the absence of target task samples," in *Proc. Int. Conf. Auton. agents multiagent Syst.*, 2015, pp. 725–733.

[38] S. Li, F. Gu, G. Zhu, and C. Zhang, "Context-aware policy reuse," in *Proc. 18th Int. Conf. Auton. Agents MultiAgent Syst.*, 2019, pp. 989–997.

[39] A. Agostinelli, J. Uijlings, T. Mensink, and V. Ferrari, "Transferability metrics for selecting source model ensembles," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 7926–7936.

[40] F. Belletti, D. Haziza, G. Gomes, and A. M. Bayen, "Expert level control of ramp metering based on multi-task deep reinforcement learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 4, pp. 1198–1207, Apr. 2018.

[41] J.-H. Cho, V. Jayawardana, S. Li, and C. Wu, "Model-based transfer learning for contextual reinforcement learning," in *Proc. 38th Conf. Neural Inf. Process. Syst.*, 2024, pp. 1–41.

[42] X.-S. Wei, C.-L. Zhang, L. Liu, C. Shen, and J. Wu, "Coarse-to-fine: A RNN-based hierarchical attention model for vehicle re-identification," in *Proc. Asian Conf. Comput. Vis.* Springer, 2018, pp. 575–591.

[43] Y. Wang et al., "Coarse-to-fine: Progressive knowledge transfer-based multitask convolutional neural network for intelligent large-scale fault diagnosis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 2, pp. 761–774, Feb. 2023.

[44] A. Hasan, N. Chakraborty, C. Wu, and K. Driggs-Campbell, "Towards co-operative congestion mitigation," 2023, *arXiv:2302.09140*.

[45] C. C. Macadam, "Understanding and modeling the human driver," *Veh. Syst. Dyn.*, vol. 40, no. 1–3, pp. 101–134, 2003.

[46] M. Bando, K. Hasebe, A. Nakayama, A. Shibata, and Y. Sugiyama, "Dynamical model of traffic congestion and numerical simulation," *Phys. Rev. E*, vol. 51, no. 2, pp. 1035–1042, 1995.

[47] Y. Sugiyama, "Optimal velocity model for traffic flow," *Comput. Phys. Commun.*, vol. 121, pp. 399–401, 1999.

[48] X. Liang, S. I. Guler, and V. V. Gayah, "Joint optimization of signal phasing and timing and vehicle speed guidance in a connected and autonomous vehicle environment," *Transp. Res. Rec.*, vol. 2673, no. 4, pp. 70–83, 2019.

[49] S. Liu, W. Zhang, X. Wu, S. Feng, X. Pei, and D. Yao, "A simulation system and speed guidance algorithms for intersection traffic control using connected vehicle technology," *Tsinghua Sci. Technol.*, vol. 24, no. 2, pp. 160–170, 2018.

[50] X. Ma, X. Hu, S. Schweig, J. Pragalathan, and D. Schramm, "A vehicle guidance model with a close-to-reality driver model and different levels of vehicle automation," *Appl. Sci.*, vol. 11, no. 1, p. 380, Jan. 2021.

[51] Z. Wang, M. Abdel-Aty, L. Yue, J. Zhu, O. Zheng, and M. H. Zaki, "Investigating the effects of human–machine interface on cooperative driving using a multi-driver co-simulation platform," *IEEE Trans. Intell. Veh.*, vol. 9, no. 1, pp. 2808–2821, Jan. 2024.

[52] F. Mannering, "An empirical analysis of driver perceptions of the relationship between speed limits and safety," *Transp. Res. Part F: Traffic Psychol. Behav.*, vol. 12, no. 2, pp. 99–106, Mar. 2009.

[53] V. Jayawardana, C. Tang, S. Li, D. Suo, and C. Wu, "The impact of task underspecification in evaluating deep reinforcement learning," in *Adv. Neural Inf. Process. Syst.*, vol. 35. Red Hook, NY, USA: Curran Associates, Inc., 2022, pp. 23881–23893.

[54] H. Wang, S. Zheng, C. Xiong, and R. Socher, "On the generalization gap in reparameterizable reinforcement learning," in *Proc. Int. Conf. Mach. Learn*. PMLR, 2019, pp. 6648–6658.

[55] C. Benjamins et al., "Contextualize me–the case for context in reinforcement learning," *Trans. Mach. Learn. Res.*, pp. 1–38, 2023.

[56] J. J. Garau-Luis et al., "Multi-objective evolution for generalizable policy gradient algorithms," in *Proc. Int. Conf. Learn. Representations 2022 Workshop Generalizable Policy Learn. Phys. World*, 2022, pp. 1–23.

[57] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Phys. Rev. E*, vol. 62, no. 2, pp. 1805–1824, Aug. 2000.

[58] P. A. Lopez et al., "Microscopic traffic simulation using SUMO," in *Proc. 21st IEEE Int. Conf. Intell. Transp. Syst*., IEEE, 2018, pp. 2575–2582.

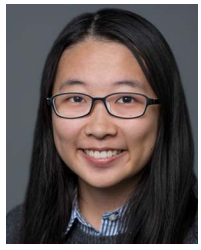
[59] A. Reuther et al., "Interactive supercomputing on 40,000 cores for machine learning and data analysis," in *Proc. IEEE High Perform. Extreme Comput. Conf*., IEEE, 2018, pp. 1–6.

[60] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn*. PMLR, 2015, pp. 1889–1897.

[61] Y. Sugiyama et al., "Traffic jams without bottlenecks—experimental evidence for the physical mechanism of the formation of a jam," *New J. Phys.*, vol. 10, no. 3, pp. 1–7, 2008.

Jung-Hoon Cho (Graduate Student Member, IEEE) received the B.S. and M.S. degrees in civil and environmental engineering from Seoul National University, Seoul, South Korea, in 2020 and 2022, respectively. He is currently working toward the Ph.D. degree in CEE with the Department of Civil and Environmental Engineering and the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA, USA.

His research interests include the convergence of transportation systems and machine learning. This involves delving into smart infrastructure, such as autonomous driving systems, traffic control in scenarios of mixed autonomy, and urban data intelligence through the application of machine learning.



Sirui Li received the B.S. degree in computer science and mathematics from Washington University, St. Louis, MO, USA, in 2019, and the Ph.D. degree in social and engineering system and statistics from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2025.

Her research interests include the areas of machine learning for combinatorial optimization and control analysis for transportation systems.

Jeongyun Kim received the B.S., M.S., and Ph.D. degrees in civil and environmental engineering from the Department of Civil and Environmental Engineering, Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 2014, 2016, and 2021, respectively.

She was a Postdoctoral Researcher with Massachusetts Institute of Technology, Cambridge, MA, USA. She is currently an Assistant Professor with the Department of Mechanical and Automotive Engineering, Seoul National University of Science and Technology, Seoul, South Korea. Her research is in the area of modeling urban mobility and developing solutions to the challenges related to the autonomous urban mobility system. Her research interests include urban mobility analytics using large-scale movement data and modeling autonomy control system using machine learning.

Cathy Wu (Member, IEEE) received the B.S. and M.Eng. degrees in electrical engineering and computer science from Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 2012 and 2013, respectively and the Ph.D. degree in electrical engineering and computer science from University of California, Berkeley, Berkeley, CA, USA, in 2018.

She was a Postdoctoral Researcher with Microsoft Research, Redmond, WA, USA. She is currently the Class of 1954 Career Development Associate Professor with MIT in LIDS, CEE, and IDSS. Her research focuses on AI for engineering, particularly to accelerate R&D for the public interest and in civil infrastructure. In recent years, her focus has been on using machine learning to tackle bottlenecks in solving control and optimization problems in mobility.

Dr. Wu was the recipient of the NSF CAREER, Ph.D. dissertation awards, and the Ole Madsen Mentoring Award. She serves on the Board of Governors for the IEEE ITSS, is a Program Co-chair for RLC 2025, and is an Associate Editor (or equivalent) for International Conference on Machine Learning, Conference and Workshop on Neural Information Processing Systems, International Conference on Robotics and Automation, and Transportation Research Part C. She is also the inaugural Chair and Co-Founder of the REproducible Research In Transportation Engineering (RERITE) Working Group.