Enhancing Autonomous Mobility on Demand Systems: A Hierarchical Repositioning Approach Integrating Regional-level and Route-level Decision

Taijie Chen, *Graduate Student Member, IEEE,* Jingyun Liu, Siyuan Feng, Jiandong Qiu, Jintao Ke

***Abstract*—Autonomous mobility-on-demand (AMoD) systems face persistent challenges due to the spatio-temporal mismatch between vehicle supply and passenger demand, which results in low fulfillment rates and inefficient fleet utilization. Existing repositioning strategies primarily follow two paradigms. Region-level approaches direct idle vehicles to high-demand areas using coarse-grained policies but often fail to provide effective guidance within the target region. In contrast, route-level methods offer fine-grained control by generating paths on the road network, yet they frequently lack global planning and overlook broader supply-demand dynamics. To address the limitations of both paradigms, we propose a novel *top-to-bottom repositioning (T2BR)* framework that hierarchically integrates decision-making at multiple levels. At the regional level, reinforcement learning is employed to optimize inter-regional movements of idle vehicles based on long-term platform objectives. At the route level, Monte Carlo Tree Search is utilized to generate context-aware paths that facilitate efficient passenger pickups within target regions. This hierarchical structure allows for dynamic, adaptive, and spatially coordinated repositioning decisions. Comprehensive evaluations using real-world operational data from Manhattan demonstrate that the proposed T2BR framework significantly improves key performance metrics, including order fulfillment rate, platform revenue, and vehicle utilization, when compared to existing baseline methods. These results highlight the effectiveness of our approach in enhancing the operational efficiency of AMoD systems.**

***Index Terms*—Autonomous mobility-on-demand systems, vehicle repositioning, reinforcement learning, and Monte Carlo Tree Search.**

I. Introduction

THE rise of mobility-on-demand (MoD) services such as Uber and Lyft has transformed urban transportation by offering flexible, point-to-point travel options. With the advancement of autonomous vehicle (AV) technologies, autonomous MoD (AMoD) services have emerged as the next evolutionary step. As of April 2024, China's Apollo Go has completed over 6 million autonomous ride-hailing services and is piloting operations in 11 cities. In the United States, over 100,000 users have registered for Waymo's AV service.

AMoD is a subtype of MoD, which also requires specialized algorithms for driver-passenger matching [1], [2], pricing [3], [4], and vehicle repositioning [5]–[8]. Studies show that up to 40% of passenger requests may go unserved [9], and vehicles may spend nearly 50% of their time cruising without passengers [10]. In this study, we aim to design a repositioning algorithm to address the spatio-temporal mismatch between passenger demand and vehicle supply. Efficient repositioning, which involves moving empty vehicles from a low-demand region to a high-demand region, is crucial for enhancing system performance. This includes optimizing driver occupancy rate (OR), improving order fulfillment rates (FR), and increasing platform revenue (PR) [5], [6], [11].

Existing repositioning methods often employ a region-level repositioning paradigm in which the service area is partitioned into discrete grid cells, such as rectangles or hexagons [12]–[14]. Under this paradigm, idle vehicles are assigned to a target region, typically the one with the highest anticipated demand, and routed there via the shortest path. Although this strategy simplifies decision-making and supports large-scale optimization, it exhibits two significant drawbacks. First, it overlooks intra-region dynamics. Upon arrival at the designated cell, usually at its centroid or along a boundary road, the vehicle receives no further navigational guidance for exploring the local street network. Consequently, it either remains idle or conducts unguided searches, resulting in prolonged idle times and inefficient pickups, especially in large or irregularly shaped regions (see Fig. 1, left). Second, the approach applies identical actions to all vehicles within the same cell, ignoring each vehicle's exact position and real-time context; this uniform treatment precludes the formulation of individualized repositioning strategies and ultimately degrades overall system performance.

To overcome these limitations, some researchers have proposed route-level repositioning strategies that directly operate within the road network, guiding vehicles along optimal paths based on traffic patterns and demand distributions (see Figure 1, center). For instance, Garg et al. [15] employ Monte Carlo Tree Search (MCTS) to identify high-reward road segments. However, pure route-level approaches encounter three main challenges: 1) They lack comprehensive global planning, which may lead to suboptimal or redundant routes. For instance, in cases where there is zero demand in a particular

This work was supported by the Smart Traffic Fund (PSRI/29/2201/PR) of the Hong Kong SAR Government, China, and the General Research Fund (GRF) of the Research Grants Council of Hong Kong, China (HKU15209121; PolyU15207424). (Corresponding author: Siyuan Feng.)

Taijie Chen and Jintao Ke are with the Department of Civil Engineering, the University of Hong Kong, Hong Kong, China (email: ctj21@connect.hku.hk, kejintao@hku.hk).

Jingyun Liu is with the Department of Electrical and Computer Engineering, Carnegie Mellon University, US (email: jeanliu@andrew.cmu.edu).

Siyuan Feng is with the Department of Aeronautical and Aviation Engineering, The Hong Kong Polytechnic University, Hong Kong, China (email: siyuan.feng@polyu.edu.hk).

Jiandong Qiu is with the Shenzhen Urban Transport Planning Center Co., Ltd, Shenzhen, China (email: Qiujd@sutpc.com).

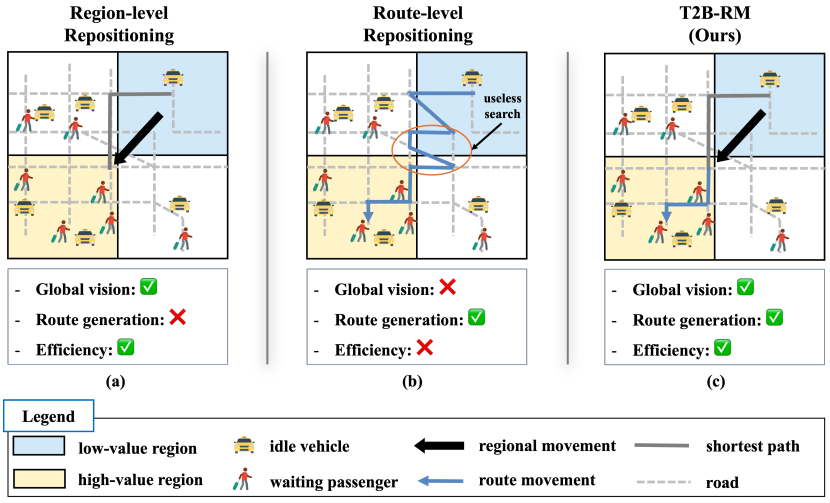


Fig. 1: Comparison of region-level repositioning, route-level repositioning, and our proposed T2BR.

region, vehicles should relocate to other regions instead of searching for passengers there. 2) These strategies can be computationally intensive due to their recursive search processes. 3) They often do not account for temporal dynamics, such as fluctuating demand peaks.

To overcome the drawbacks of both kinds of approaches aforementioned, we propose a hierarchical repositioning framework, termed the top-to-bottom repositioning (T2BR) model, which integrates region-level and route-level decisions to enhance system-wide efficiency (Fig. 1, right). Our approach offers several advantages. First, it incorporates a global decision-making perspective. When there is a significant imbalance between supply and demand, vehicles are efficiently guided via the shortest path to high-demand target regions. This mechanism reduces unnecessary local exploration, effectively serving as a pruning strategy for the MCTS algorithm. Conversely, when a vehicle is already located in a high-demand area, the system enables targeted intra-region exploration to quickly identify potential passengers. Second, the framework supports the generation of heterogeneous searching trajectories for vehicles within the same region. By assigning individualized paths based on vehicle context and local demand conditions, T2BR mitigates intra-region competition and improves overall system efficiency.

We evaluate the effectiveness of the proposed T2BR framework using real-world data from Manhattan. Experimental results demonstrate substantial improvements in key performance metrics, including FR, PR, and DR, when compared to state-of-the-art baseline methods. The main contributions of this work are summarized as follows:

- We identify key limitations in existing region-level and route-level repositioning approaches and introduce a unified hierarchical decision framework to overcome them.
- We develop T2BR that integrates reinforcement learning (RL) (for region-level repositioning) with MCTS (for route-level repositioning), enabling dynamic, spatially aware, and temporally adaptive repositioning.
- We conduct extensive experiments based on real operational data from Manhattan, demonstrating that T2BR consistently outperforms several strong baselines.

The remainder of this paper is organized as follows. Section II reviews the related literature. Section III formally defines the problem setting. Section IV details the T2BR. Section V describes the experimental setup. Section VI presents and discusses the experimental results. Section VII provides sensitivity analyses. Finally, Section VIII concludes the paper.

II. Literature Review

Efficient vehicle repositioning is a core operational task in AMoD platforms, with significant implications for fleet utilization, passenger waiting time, and system-wide efficiency. A well-designed repositioning strategy can reduce the fleet size by up to 30% [16] and cut down vehicle travel distance by 40% [17]. In recent years, a growing body of research has explored various repositioning strategies, from region-level repositioning to route-level repositioning and hierarchical repositioning. In this section, we critically review the literature from three perspectives: region-level, route-level, and hierarchical repositioning, focusing on their core objectives, technical foundations, and key limitations.

A. Region-level Vehicle Repositioning

Region-level strategies discretize the operational area into spatial units (e.g., grid cells, hexagons) [12], [18], [19] and aim to mitigate spatio-temporal mismatch between supply and demand by repositioning idle vehicles across regions. Optimization goals vary across studies, including revenue maximization [20], [21], cost minimization [22], driver income enhancement [23], and balancing utilization with idle mileage [18].

Algorithmic approaches range from rule-based heuristics and linear programming [24] to more recent RL models, which offer adaptability under dynamic and uncertain conditions [25], [26]. For example, Xie et al. [27] proposed a multi-agent deep RL framework for jointly managing human-driven and autonomous vehicles in shared environments. Similarly,

i-Rebalance [11] incorporates driver preferences into RL to personalize repositioning guidance and improve driver compliance. Lyu et al. [8] utilize a large language model to make joint decisions for order dispatching and repositioning.

Despite their merits, region-level methods face notable challenges: (1) They are highly dependent on demand forecasts, which may be unreliable in volatile contexts; (2) Centralized control frameworks can struggle with scalability and fail to adapt to local variations; (3) The spatial discretization introduces a granularity barrier that can hinder responsiveness in fast-changing, fine-scale environments.

B. Route-level Vehicle Repositioning

Route-level repositioning focuses on fine-grained control of idle vehicle movements across road segments, often supplementing regional strategies. Here, vehicles dynamically select their next road segment at decision points based on real-time signals [28], [29], aiming to improve the likelihood of ride-matching while reducing idle cruising time [30].

Earlier approaches utilized static features (e.g., road length, historical demand) [30], whereas newer methods integrate dynamic indicators such as predicted order density, traffic congestion, and weather conditions into route-level decision models [15]. However, real-time route planning at the city scale remains computationally intensive. Moreover, many models treat road segments independently, neglecting spatial dependencies and system-level feedback effects. These myopic and reactive policies also often lack coordination with broader repositioning goals, limiting their strategic effectiveness.

C. Hierarchical Vehicle Repositioning

Recent studies have explored hierarchical repositioning frameworks to bridge the gap between global planning and local responsiveness. These frameworks coordinate decisions across multiple spatial or temporal scales. For instance, Zhu et al. [7] integrate model predictive control with coverage control to align regional flow with vehicle-level movements, while Beojone et al. [31] proposed a three-layer architecture leveraging macroscopic flow models, mid-level dispatch, and microscopic vehicle guidance.

These hierarchical approaches show promise in balancing system-wide objectives with real-time adaptability. However, limitations remain: (1) Many rely on static or pre-specified demand patterns, lacking robust mechanisms for online adaptation; (2) Feedback between hierarchical levels is often implicit or hand-engineered, limiting flexibility in dynamic environments; (3) Coordination mechanisms are frequently heuristic and not learned, impeding optimality under uncertainty.

In summary, while hierarchical frameworks offer a promising direction for coordinating repositioning decisions across spatial and temporal scales, current approaches still face key challenges, including limited real-time adaptability, reliance on handcrafted coordination mechanisms, and a lack of integration between global and local policies. These limitations motivate the development of a more adaptive and scalable hierarchical strategy that can better balance long-term planning with short-term responsiveness.

Building upon the above review and identified gaps, we next formalize the hierarchical vehicle repositioning problem addressed in this study.

III. Problem Formulation

In this section, we provide some definitions and formulate the region-level and route-level repositioning problems.

Definition 1 (*Intersection*). An intersection denoted as $n, n \in \mathcal{N}$ refers to a crossroad within the road network. It encompasses longitude, latitude, and a unique ID, which can be represented as $n.lon$, $n.lat$, and $n.id$ respectively.

Definition 2 (*Road Segment*). A road segment, represented as $e, e \in \mathcal{E}$, signifies a stretch of roadway in the transportation network. It is the edge between any two accessible intersections. Each road segment is characterized by ID of starting intersection, the ID of an ending intersection, its length, and a unique ID, which can be denoted as $e.start\_id$, $e.end\_id$, $e.length$, and $e.id$ respectively.

Definition 3 (*Road Network*). A road network is defined as $\mathcal{G} = (\mathcal{E}, \mathcal{N})$, where $\mathcal{N}$ represents the set of all nodal points (i.e., intersections) within a specified region, and E denotes the collection of road segments. The network is structured such that any $e_{ij} \in \mathcal{E}$ is equivalent to $n_j \in \mathcal{N}$ being the next reachable node from $n_i \in \mathcal{N} \backslash \{n_j\}$.

Assume that there is a need to provide automated, on-demand transportation services within a specified time frame for a designated area denoted as $\mathcal{G} = (\mathcal{E}, \mathcal{N})$. Initially, the service region is divided into N non-overlapping zones, which may be rectangular or hexagonal. Each zone is represented as $\mathcal{G}_i = (\mathcal{E}_i, \mathcal{N}_i)$, where $\mathcal{E}_i$ and $\mathcal{N}_i$ respectively signify the edges and network nodes within the i-th zone. Further, it is assumed that the period of study spans from t_{init} and t_{end}, with t_{init} and t_{end} representing the start and end times of the study period, respectively. Each simulation time step is represented as $t, t \in \mathcal{T}$. The study period is divided into several intervals of duration Δt, resulting in a total of $T = (t_{\text{end}} - t_{\text{init}})/\Delta t = ||\mathcal{T}||$ intervals.

Definition 4 (*Idle Vehicles*). Idle Vehicles refer to those that are in operational status but are still cruising without passengers, as they have not received any service requests. Let $\mathcal{V}$ denote the set of all vehicles, with $||\mathcal{V}||$ representing the total number of autonomous vehicles. $\mathcal{V}^t_{\text{idle}}$ represents the set of all vehicles at time interval $t, t \in \{1, 2, \cdots, T\}$, $||\mathcal{V}^t_{\text{idle}}||$ represents the number of available vehicles at time step t.

Because the main subject of this research is the AMoD system, we make the following assumptions:

A1. AVs strictly obey the platform's region-level decision instructions [7].

A2. AVs can make route-level decisions autonomously.

A3. AVs in the same state are homogenous [19].

A4. AVs have no preference for orders.

A5. An autonomous vehicle will not go offline, and the time it needs to refuel or recharge is considered negligible.

Definition 5 (*Waiting Passengers*). $\mathcal{O}^t_{\text{wait}}, t \in \{1, 2, \cdots, T\}$ represents the accumulated waiting passengers at time step t.

Definition 6 (*Repositioning Route*). A route is denoted as $\mathcal{P} = \{e_1, e_2, \cdots, e_n\}$ such that $\forall i, 1 \leq i \leq n-1, e_i.\text{endid} =$

e_{i+1}.startid. Idle vehicles will follow the given route, link by link, to find passengers.

P1: At time interval $t, t \in \mathcal{T}$, given $\mathcal{O}^t_{\text{wait}}$ and $\mathcal{V}^t_{\text{idle}}$, for each idle vehicle $v, v \in \mathcal{V}^t_{\text{idle}}$ located in region $i, i \in \mathcal{R}$, determine move vehicle v to which region $j, j \in \mathcal{R}_i$. After arriving at the target region, determine the repositioning route $\mathcal{P}$.

Based on this problem formulation, we develop a hierarchical methodology to solve the vehicle repositioning task.

IV. Methodology

To solve **P1**, we present the T2BR model (Fig. 2). The design motivation of this method is demonstrated in Section IV-A. This hierarchical method for vehicle repositioning is divided into two primary layers: the upper layer utilizes RL to allocate a target region to each idle vehicle (detailed in Section IV-B); the lower layer then generates an optimal route within the assigned target region using MCTS, which is detailed in Section IV-B.

A. Design Motivation

While both RL and MCTS are commonly used algorithms, their integration is non-trivial in the context of vehicle repositioning for the AMoD system. Inspired by successful applications in the game of Go [32] and language models [33], we design a hierarchical decision framework where RL decides region-level movements and MCTS performs fine-grained intra-region path search. This design is motivated by the following considerations:

- RL provides global guidance and filters out low-value regions, thus pruning the MCTS search space and improving efficiency.
- MCTS captures short-term uncertainty and enables local exploitation within regions with high pickup potential.
- The hierarchical structure balances exploration and exploitation, avoids local optima, and enables adaptive decision-making under spatial-temporal dynamics.

This layered structure allows each component to operate in its area of strength while compensating for the other's limitations.

B. Upper layer

In the upper layer of the T2BR, we define the problem as a Markov Decision Process (MDP), defined by a tuple (N, S, A, R, P, γ), where N, S, A, R, P, γ are the number of agents, sets of states, joint action space, transition probability functions, reward functions, and a discount factor respectively.

Agent: We treat each idle vehicle as an agent. However, there are numerous vehicles in one city, which means that training a large number of agents is very time-consuming and requires a lot of resources. To solve this problem, based on **A3**, we treat vehicles in the same grid are homogeneous. Although the count of homogeneous agents remains constant at N (where N represents # of grids), the number of heterogeneous agents N_t at time step t is time-dependent.

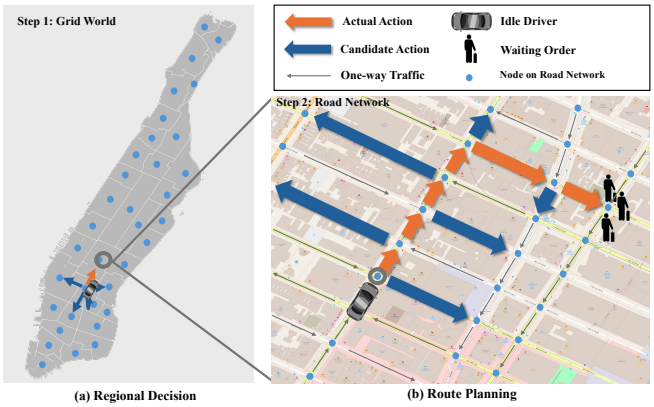


Fig. 2: Illustration of top-to-bottom repositioning framework.

State: At each time step t, the agent draws a global state $\boldsymbol{s}_t \in S$ that captures the spatial distributions of available vehicles and orders (i.e., the number of available vehicles, orders in each grid, and the current time step t (encoded using one–hot encoding)). We define the state of an individual agent i, denoted as s^i_t, is defined by combining the global state with the specific grid location of the agent: $s^i_t = [\boldsymbol{s}_t, g_i]$, where g_i represents the one-hot encoded grid ID. It is important to note that agents located in the same grid share an identical state s^i_t.

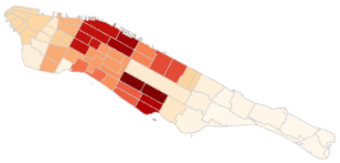


Fig. 3: Action space. Manhattan is segmented into 35 neighborhood regions based on the 2020 Neighborhood Tabulation Areas. We merge one and remove two, therefore there are 32 regions in our study.

Action: $\boldsymbol{a}_t \in A = A_1 \times A_2 \times \cdots \times A_N$: a joint action $\boldsymbol{a}_t = \{a^i_t\}^N_1$ instructing the repositioning strategy of all available vehicles at time t. The action space A_i of an individual agent specifies where the agent i is able to arrive at the next time, which gives a set of seven discrete actions denoted by $\{k\}^{32}_{k=1}$ (as shown in Fig. 3).

Reward: $R_i \in R = S \times A \to \mathbb{R}$. In our hierarchical model, for agent i at time step t, the reward consists of two parts: one is the reward for region-level movement, denoted as $r^t_{u,i}$, which guides vehicles to move towards regions with higher potential value, and the other is the expected reward for route-level movement in the target area, represented as $r^t_{l,i}$, which evaluates the expected value of exploring specific road segments within the target region. The total reward is represented as r^t_i, and r^t_i can be calculated as follows:

$$r^t_i = \beta r^t_{u,i} + (1-\beta) r^t_{l,i}, \tag{1}$$

where $\beta \in [0, 1]$ is the coefficient to combine region-level reward and route-level reward.

- Region-level reward $r^t_{u,i}$: This reward at the time step t for a vehicle taking action $a^i_t, a^i_t \in \{k\}^{32}_{k=1}$ is designed to guide vehicles to move to regions with higher potential value. It is based on the average revenue accrued from vehicles who move to the same grid and can be calculated as follows:

$$r^t_{u,i,a^i_t} = \sum_{o \in O_{a^t_i}} o.p / n^t_{a^i_t}, \tag{2}$$

  where $O_{a^t_i}$ represents the orders matched by vehicles that take action a^t_i at time step t, $o.p$ is the revenue that the vehicle can earn after completing order o, and $n^t_{a^t_i}$ is the number of vehicles taking action a^t_i at time step t.
- Route-level reward $r^t_{l,i}$: This reward at the time step t for a vehicle taking action a^t_i evaluates the expected value of exploring specific road segments within the target region. It can be calculated as follows:

$$r^t_{l,i,a^i_t} = \mathbb{E}\left[\mathcal{E}_{a^i_t}\right] = \frac{1}{||\mathcal{E}_{a^i_t}||} \sum_{e \in \mathcal{E}_{a^i_t}} e.p, \tag{3}$$

  where $\mathcal{E}_{a^i_t}$ is the set of edges (road segments) in the target region after taking action a^i_t, e is a road segment, and $e.p$ is the expected revenue that the vehicle can earn after traversing segment e.

At time step t, the i-th agent attempts to maximize its own expected discounted return: $\mathbb{E}\left[\sum_{k=t}^{T} \gamma^{k-t} r^k_i\right]$.

State transition probability: State Transition $p(\boldsymbol{s}_{t+1}|\mathbf{s}_t, a) \in P : S \times A \times S \rightarrow [0,1]$: It gives the probability distribution of taking action a given state $\boldsymbol{s}_t$, to a certain next state $\boldsymbol{s}_{t+1}$.

Specifically, we develop the multi-agent Actor-Critic (MAAC) algorithm to solve the MDP problem. The MAAC consists of two parts. The first one is the centralized value network (i.e., Critic) shared by all agents. The second one is the policy network; the joint action of all agents is the input to the policy network, which explicitly establishes the coordination among agents.

C. Lower layer

Upon selecting a target grid using the MAAC policy, the agents proceed to the lower layer, where MCTS is utilized for route generation. This stage concentrates on optimizing the sequence of order placements to ensure efficiency and timeliness.

MCTS is an algorithm used to make optimal decisions in complex decision-making environments, particularly useful in games and planning problems [34]. The algorithm proceeds by building a search tree incrementally, node by node, based on the outcomes of numerous simulations. These simulations are used to estimate the potential success of moves from a given state.

In our study, we model each road network node as a state, with connecting road segments (referred to as road network edges) representing the available actions at that state. The MCTS process is structured into four distinct stages as follows:

1) Selection: : During this initial stage, a vehicle at a current road network node selects a road segment from the action space, which is all the available edges given a road node. This decision is based on the upper confidence bound (UCB) policy [35], which assesses the suitability of the segment given the current state (Equation 4). The specifics of the scoring mechanism will be detailed in the UCB section.

$$\pi^*(t) = \arg\max_{e \in \mathcal{E}_n} UCB_e(t), \tag{4}$$

2) Expansion: : If the node reached by the vehicle has never been visited before, it is considered a leaf node. For a leaf node in the search tree, it is expanded to include all connected road segments $\mathcal{E}_n$.

3) Simulation: : This stage involves a rollout simulation starting from the newly expanded node. The simulation proceeds until either the vehicle is successfully matched to an order or a predefined search depth is reached, whichever occurs first.

4) Backpropagation: : Starting from the terminal state achieved during the simulation, the results are propagated backward through the path taken. Throughout this stage, the scores of the actions selected during the route are updated based on their outcomes.

These stages collectively enable the dynamic generation of optimized routes within the road network, leveraging probabilistic path exploration to enhance decision-making in real-time.

UCB: MCTS is renowned for its effective balance between the exploration of new possibilities and the exploitation of advantageous moves. This balance is achieved through a policy known as the UCB, which is applied during the selection step. The UCB for each action is calculated using Equation 5.

$$UCB_i(t) = \overline{\mu}_i(t) + c\sqrt{\frac{\ln t}{n_i(t)}} + \epsilon, \tag{5}$$

where $\overline{\mu}_i(t)$ refers to the average outcome (or utility) of taking an action from node i at time t, based on past simulations. This is equivalent to $Q(s,a)$, where s denotes the state and a the action in the RL context. $n_i(t)$ represents visit counts in the equation, which tracks the number of times node i has been visited at time t. $c\sqrt{\frac{\ln t}{n_i(t)}}$ is the exploration term, which is the part of the UCB formula that encourages exploration of less-visited nodes.

In terms of calculating $\overline{\mu}_i(t)$, after a simulation rollout, a trajectory is determined to navigate the vehicle from the target grid centroid/boundary to a potential passenger. There are two possible outcomes in the simulator: the vehicle either successfully picks up a customer or fails to do so. The goal is to generate a route that maximizes fare revenue and minimizes travel costs. Minimizing travel costs not only reduces expenses but also serves as a proxy for the customer's utility level, as shorter travel times lead to reduced waiting periods for pickups. The rewards are allocated to each road segment along the route as calculated by Equation 6.

$$\overline{\mu}_i(t) = \begin{cases} p_o - c \cdot \sum_{e \in \mathcal{P}_i(t)} d_e, & \text{if matched with an order } o; \\ -c \cdot \sum_{e \in \mathcal{P}_i(t)} d_e, & \text{if failed to match.} \end{cases} \tag{6}$$

The definitions are as follows:

- p_o: The revenue earned for an order o.
- c: Fuel cost per kilometer. A representation of the cost of the repositioned ride.
- $\mathcal{P}_i(t)$: The MCTS trajectory for agent i at time t. A trajectory consists of connected road segments.
- d_e: The distance of the road segment (an edge in the road network) in kilometers.

In the following section, we present the experimental design, covering the data sources, simulation environment, baseline comparisons, and performance metrics that will be used to rigorously evaluate our method.

V. Experiment

A. Settings

We develop and test our algorithms using the simulator developed by [36]. The simulator settings are detailed in Appendix A, while the hyperparameter settings for the neural network are provided in Appendix A. The training and simulation phases were executed on a server equipped with an Intel(R) Core(TM) i9-10900X CPU, 128 GB RAM, and two Nvidia 4090 GPUs.

B. Data

The dataset contains 65,955 ride-hailing orders from Manhattan, captured between 5:00 AM and 10:00 AM on Monday, May 4, 2015. The focus is on the Manhattan borough's road network, which comprises 4,474 nodes and 9,682 edges. Orders with origins or destinations outside Manhattan were excluded. We adopted different sampling ratios to conduct algorithm testing in various supply and demand scenarios. The dataset details for different scenarios are presented in Table I. The total number of orders per hour during the study period is shown in Fig. 4. To ensure that the order data used for training and validation are different, we used a random seed of 42 to sample the orders during the training process. For the validation process, we conducted five sets of experiments with random seed numbers set to 42, 2021, 2022, 2023, and 2024.

TABLE I: Dataset Statistics.

Scenario	# of Vehicles	# of Orders	Sample rate
Low Density	100	3,296	5%
Medium Density	100	6,589	10%
High Density	100	32,967	50%

C. Baseline Models

- **Random:** The relocation is executed by uniformly selecting available actions randomly.
- **Stay:** Empty vehicles stay in their current area and do not move to surrounding areas.

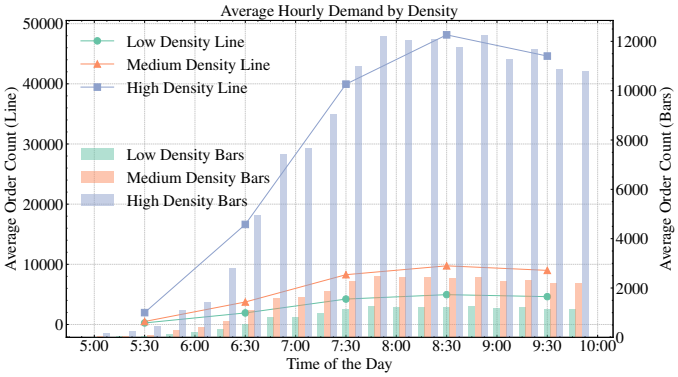


Fig. 4: The temporal distribution of the demand under different supply-demand scenarios. Our study period ranges from 5:00 am to 10:00 am. The line chart represents the total demand accumulated in the first and second halves of each hour, corresponding to half-hour intervals. In contrast, the bar chart illustrates the total demand at 15-minute intervals.

- **Greedy-1:** The empty vehicle will select an action a from the optional action set A. After taking action a, it can go to the area with the highest density demand in the empty vehicle reposition candidate area.
- **Greedy-2:** The empty vehicle will select an action a from the optional action set A. After taking action a, it can go to the area with the highest demand/supply ratio in the empty vehicle reposition candidate area.
- **Mathematical Model (MP):** Let $\mathcal{R}$ represents the set of regions in the study area, and $|\mathcal{R}|$ denotes the total number of regions. The binary decision variable x_{ij}, for all $i, j \in \mathcal{R}$, indicates whether vehicles are dispatched from region i to region j. The cost associated with dispatching vehicles from region i to region j is denoted by c_{ij}, for all $i, j \in \mathcal{R}$. Each region j has a specified demand d_j and a supply of vehicles s_j for all $j \in \mathcal{R}$. The model aims to minimize the total dispatching cost plus the disparity between demand and supply across regions after dispatching. This is formulated as the objective function:

$$\min \sum_{i \in \mathcal{R}} \sum_{j \in \mathcal{R}} c_{ij} x_{ij} + \sum_{j \in \mathcal{R}} \left| d_j - \left(s_j + \sum_{i \in \mathcal{R}} x_{ij} - \sum_{k \in \mathcal{R}} x_{jk} \right) \right| \tag{7a}$$

$$\sum_{j \in \mathcal{R}} x_{ij} \leq s_i, \quad \forall i \in \mathcal{R}, \tag{7b}$$

$$x_{ij} \geq 0, \quad \forall i, j \in \mathcal{R}, \tag{7c}$$

$$x_{ij} \in \mathbb{Z}^+, \quad \forall i, j \in \mathcal{R}. \tag{7d}$$

- **MCTS [15]:** We reproduce a route-level strategy where idle vehicles generate cruising trajectories by iteratively selecting road segments using the MCTS algorithm.
- **DA2C:** A decentralized region-level repositioning baseline that employs the Advantage Actor-Critic algorithm for decision-making.

- **D2QN:** A decentralized region-level repositioning method based on DQN, where the agent learns value-based policies for vehicle allocation across zones.
- **D3QN:** A variant of the decentralized region-level approach using DDQN to mitigate Q-value overestimation and improve policy stability.
- **Hierarchical Control (HC) [6]:** We re-implement the two-layer framework that combines inter-region rebalancing with intra-region coverage control.

D. Metrics

We define several performance metrics to evaluate the algorithms in this study, which include PR, FR, OR, and delivery rate (DR), as well as waiting time (WT), matching time (MT), and pickup time (PT). And we partition these metrics into time-related and non-time-related.

- **PR:** Total platform revenue over the testing periods, with US dollar as a unit, which can be calculated by $\sum_{o \in \mathcal{O}_{\text{served}}} o.\text{revenue}$.
- **FR:** Fulfillment rate of all orders, which can be calculated by $|\mathcal{O}_{\text{served}}| \,/\, |\mathcal{O}|$, where $|\mathcal{O}_{\text{served}}|$ and $|\mathcal{O}|$ indicate the number of served orders and the number of total orders in the study period respectively.
- **OR:** The average occupancy rate of vehicles, can be calculated by the total time that vehicles are under-occupied status (including pickup and delivery stages) divided by the total time that vehicles are under operation status.
- **Delivery rate (DR):** Delivery rate can be calculated by the total time that vehicles are under delivery status divided by the total time that vehicles are under operation status.
- **Matching time (MT):** Average waiting time for vehicle-passenger matching of all served orders.
- **Pickup time (PT):** Average pickup time for picking up passengers after matching of all served orders.
- **Waiting time (WT):** Average waiting time (including waiting time for vehicle-passenger matching and waiting time for picking up passengers) of all served orders.

With these experimental settings in place, we next present and analyze the empirical results obtained from comprehensive evaluations.

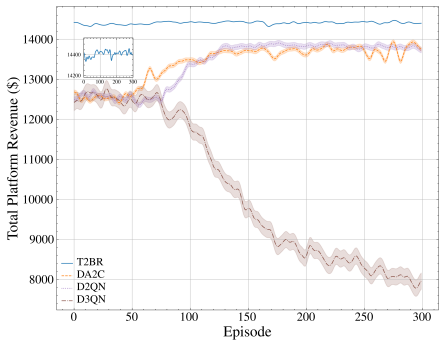
VI. Results and Discussion

A. Training

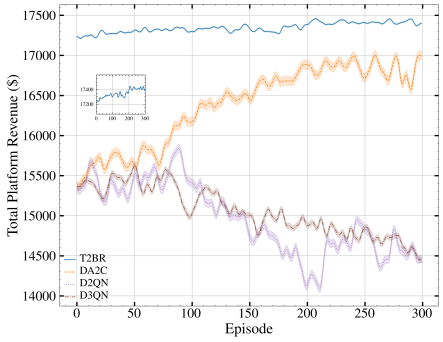
In this study, we conducted comprehensive experiments across three scenarios with varying supply and demand. Fig. 5 presents the reward curves during the training process across all scenarios. As shown, T2BR consistently demonstrates a superior initial value in all cases. Furthermore, even after RL converges, T2BR continues to outperform it.

B. Evaluation

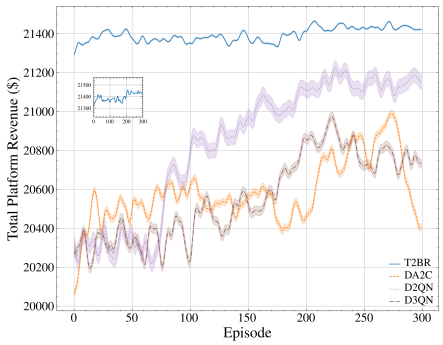
Table II summarizes the performance of all evaluated methods under different demand density scenarios. Several important patterns and insights can be observed:



(a) Low density



(b) Medium density



(c) High density

Fig. 5: The reward curves are plotted against training epochs. Subfigures (a), (b), and (c) illustrate the experimental results for low, medium, and high-density scenarios, respectively.

1) Performance of Baselines: RL baselines (D2QN/D3QN, DA2C) and heuristic approaches (MP, HC) generally perform well, but their advantages are less pronounced in sparse environments. In low-density scenarios, methods like D3QN suffer severe performance degradation, with PR, FR, and OR dropping dramatically, and WT and MT becoming much longer. This is consistent with theoretical expectations: in sparse-reward environments, DDQN's value underestimation leads to insufficient exploration and suboptimal policies, while D2QN, due to its optimistic estimation bias, encourages more aggressive exploration and achieves better outcomes. Heuristic and greedy methods (MP, HC, Greedy-1, Greedy-2) remain competitive in simpler scenarios, reflecting that "local optimum" strategies can be effective when the environment is less challenging.

TABLE II: Experiment results.

Scenario	Model	PR ($) ↑	FR (%) ↑	OR (%) ↑	DR (%) ↑	WT (s) ↓	MT (s) ↓	PT (s) ↓
Low Density	Random	11060.2	59.1	49.1	42.2	151.9	48.6	103.3
	Stay	13929.3	65.6	62.4	53.0	137.5	28.8	108.7
	Greedy-1	12962.4	61.2	57.9	49.3	140.7	33.2	107.6
	Greedy-2	13049.2	61.4	58.1	49.6	143.5	36.7	106.7
	MP	14124.7	66.8	63.1	53.5	134.4	24.7	109.7
	DA2C	13956.7	65.8	62.3	53.0	137.2	29.1	108.1
	D2QN	13956.0	66.3	62.3	52.8	134.0	25.2	108.8
	D3QN	8370.5	39.4	36.7	31.9	162.9	65.5	**97.4**
	MCTS	12259.8	58.4	54.3	46.1	145.2	36.7	108.5
	HC	14124.6	66.8	63.1	53.5	137.4	27.7	109.7
	T2BR	**14526.9**	**68.8**	**64.3**	**54.7**	**131.2**	**23.7**	107.5
	Improve	+2.8%	+3.0%	+1.9%	+2.2%	2.1%	+4.0%	−10.4%
Medium Density	Random	15386.3	36.8	65.0	58.1	168.0	86.3	81.7
	Stay	15377.9	36.8	65.0	58.1	167.8	85.5	82.3
	Greedy-1	15618.0	37.1	66.2	59.1	165.1	82.1	83.0
	Greedy-2	15582.7	37.1	66.0	58.9	168.8	85.9	82.9
	MP	16977.3	40.4	72.5	64.5	164.9	79.9	85.0
	DA2C	16878.8	40.3	71.6	63.8	167.4	83.8	83.6
	D2QN	14086.6	33.5	59.8	53.6	168.9	86.9	82.0
	D3QN	15117.9	36.2	63.6	57.0	168.6	87.2	**81.4**
	MCTS	16132.2	39.0	68.5	60.6	169.6	83.1	86.5
	HC	17066.5	41.1	72.4	64.4	164.3	80.5	83.9
	T2BR	**17355.6**	**41.5**	**73.9**	**65.6**	**159.9**	**74.8**	85.1
	Improve	+1.7%	+1.0%	+1.9%	+1.7%	+2.7%	+6.4%	−4.5%
High Density	Random	20108.3	9.5	79.3	76.6	206.7	160.2	46.5
	Stay	19672.0	93.6	77.2	74.7	206.1	160.2	**45.9**
	Greedy-1	20489.6	9.8	80.4	77.7	207.1	160.8	46.3
	Greedy-2	20475.5	9.8	80.6	77.7	**203.4**	**156.8**	46.6
	MP	21135.8	9.9	84.0	81.1	205.8	159.5	46.3
	DA2C	21013.6	10.0	82.6	79.7	208.1	161.6	46.5
	D2QN	21021.8	10.0	83.1	80.0	206.6	159.3	47.3
	D3QN	20685.4	9.9	81.3	78.5	207.4	161.0	46.4
	MCTS	21213.2	10.1	83.9	80.6	209.6	161.0	48.6
	HC	21136.1	10.0	83.6	80.5	205.8	158.1	47.7
	T2BR	**21324.9**	**10.2**	**84.4**	**81.1**	205.6	157.3	48.3
	Improve	+0.5%	+1.0%	+0.5%	0%	−1.1%	−0.3%	−5.2%

Note: ↑ indicates that higher values are better for the given metric, while ↓ indicates that lower values are better. For each metric, the value of T2BR is denoted as a, while the best value among the others (excluding T2BR) is denoted as b. The improvement rate is calculated as $(a-b)/b \times 100\%$. The bold is the best, and the underlined is the second best.

2) Overall Performance and Trends of T2BR: Across all density scenarios, our proposed T2BR consistently achieves the best performance in key metrics such as PR, FR, OR, and DR, with especially significant advantages in low-density environments. In particular, T2BR's improvements are most prominent in challenging scenarios with sparse demand, where the platform faces severe supply-demand imbalance and reward signals are infrequent. For example, under low density, T2BR improves PR by 2.8% over the best baseline, while also achieving shorter WT and MT.

3) Impact of Demand Density: As demand density increases, the gap between algorithms narrows. RL-based methods (D2QN, D3QN, DA2C) become more stable and their performance improves significantly, since higher density provides richer reward signals and enables more effective policy learning. In high-density scenarios, almost all methods (except random or passive baselines) achieve close-to-optimal performance, with WT and MT slightly increasing due to the lack of vehicle supply, but overall service quality and PR reaching their highest levels. Notably, D3QN's performance recovers as the reward sparsity problem is mitigated, highlighting that its theoretical advantage in reducing Q-value overestimation is only effective when frequent rewards are available.

4) Robustness and Generalizability of T2BR: The results demonstrate the robustness and generalizability of T2BR. It achieves superior performance in all environments, particularly in low- and medium-density settings that are more representative of real-world MoD operations, where reward signals are sparse and demand-supply mismatches are common. The superior performance of T2BR in these settings highlights its capacity to learn effective policies even in the face of challenging, resource-constrained, and highly stochastic environments.

In summary, our extensive experiments confirm that T2BR offers consistently superior performance across a wide range of realistic demand scenarios, and demonstrate the relative strengths and weaknesses of existing RL and heuristic approaches under varying reward sparsity and system dynamics. These findings provide important evidence for the robustness and practical value of T2BR in MoD platforms.

To further examine the robustness and key design choices of our approach, we conduct additional sensitivity analyses as detailed in the following section.

VII. Sensitivity Analysis

A. Impact of Maximum Tree Depth at MCTS stage

As shown in Figure 6, in the MCTS stage of our model, we tested various search depths (i.e. dead number) and observed that larger search depths generally yield better results. However, the rate of improvement diminishes gradually with increased depth: there is a 3.7% improvement in PR when increasing the depth from 50 to 100, but only around a 0.4% improvement when increasing it from 100 to 150. Due to the limited improvement in evaluation metrics with increasing search depth, we set the MCTS dead number to 100 in this study.

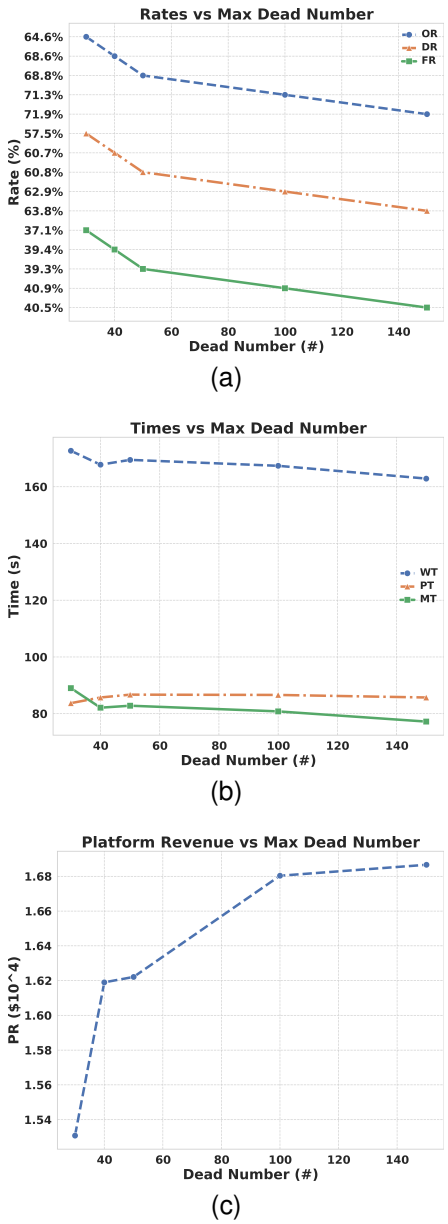


Fig. 6: A series of plots showing various metrics vs dead number for MCTS. (a) Rate-related metrics. (b) Time-related metrics. (c) PR.

B. Statistical Significance Under Representative Density Conditions

To validate the robustness of our findings, we perform paired t-tests comparing the proposed T2BR method against the DA2C and MCTS baselines under the medium supply-demand density scenario. While our experiments encompass all three density regimes (low, medium, and high), we select the medium-density setting for hypothesis testing to avoid redundancy. As shown in Table III, T2BR significantly outperforms both DA2C and MCTS across nearly all key performance metrics, with p-values well below 0.001 in most cases. This confirms that the improvements are statistically significant and not due to random variation.

The performance trends observed in the low- and high-density settings are consistent with those in the medium-density case, as reported in earlier experimental results (Section V). Therefore, we refrain from conducting repetitive significance tests under all densities and instead highlight the medium-density case as a representative illustration of T2BR's efficacy.

TABLE III: Paired t-test results for T2BR versus RL and MCTS under medium supply-demand density.

Group	Metric	t-statistic	p-value
T2BR vs. DA2C	PR (↑)	12.9196	††
	MR (↑)	16.1896	††
	OR (↑)	11.9044	††
	DR (↑)	10.1717	††
	WT (↓)	-9.3446	††
	MT (↓)	-7.2840	††
	PT (↓)	0.3957	$\sim$
T2BR vs. MCTS	PR (↑)	43.3807	††
	MR (↑)	19.3206	††
	OR (↑)	40.8545	††
	DR (↑)	33.9766	††
	WT (↓)	-16.6069	††
	MT (↓)	-10.6142	††
	PT (↓)	-3.2706	*

Note: * $p \leq 0.05$, ** $p \leq 0.01$, † $p \leq 0.005$, †† $p \leq 0.001$, $\sim$ nonsignificant.
↑ indicates that higher values are better for the given metric, while ↓ indicates that lower values are better.

C. Comparison of Inference Times

To assess the practicality of our algorithm, we measured the total time taken by three algorithms to execute repositioning decisions for all idle vehicles at each 60-second simulation interval. As shown in Table IV, T2BR reduces inference time compared to the MCTS algorithm. Although it is not as fast as the RL algorithm, an inference speed of approximately 2 seconds is still acceptable in industrial practice. Additionally, based on **A2**, all vehicles can calculate their own paths. While our implementation does not utilize multiprocessing due to resource limitations, industries can further accelerate this process by employing multiprocessing techniques.

VIII. Conclusion

In conclusion, our study further examines the crucial challenge of vehicle repositioning in the AMoD industry, which

TABLE IV: Online Inference Time for Different Algorithms

Algorithm	Low (s)	Medium (s)	High (s)
MCTS	0.981	1.138	2.026
DA2C	0.004	0.007	0.032
T2BR	0.687	0.977	1.949

is vital for balancing supply and demand across urban areas. Previous methods that emphasize minimizing travel distances or times do not necessarily align with the primary goal of maximizing vehicle engagement with potential passengers. Our proposed hierarchical framework, named T2BR, innovatively combines RL with MCTS. At the upper level, it makes region-level repositioning decisions while considering the exceptional rewards from the lower level. At the lower level, given the destination region, the system makes route-level decisions to guide vehicles. This methodology not only guides vehicles to appropriate regions but also ensures optimal route selection within those regions to enhance passenger pickup rates. Our findings from extensive experiments conducted in Manhattan demonstrate that this approach can significantly improve key performance indicators such as PR and FR across different scenarios. These results underscore the potential of our algorithm to transform operational strategies in the AMoD sector, making it a valuable tool for companies seeking to optimize their services and maximize efficiency.

Limitations and future directions: Despite the promising results, there are several limitations and real-world challenges that should be acknowledged. First, our approach does not always outperform all baselines on every metric, particularly in terms of pickup time, where some other methods can be more effective. Second, the hierarchical T2BR framework requires substantial training and inference time, placing high demands on computational resources; this may limit scalability, especially in high-demand settings where the marginal benefit over simpler approaches diminishes. Furthermore, in practical deployment, integrating region-level and route-level decisions faces additional challenges. On-vehicle computation remains a bottleneck, as autonomous vehicles must be equipped with sufficient hardware to store model parameters and perform rapid inference in real time. Reliable and low-latency communication between the central platform and distributed AVs is also crucial to enable timely and effective decision-making.

Looking forward, these limitations point to promising directions for future research. One avenue is to make the integration parameters between the upper and lower levels learnable through deep learning techniques, which could further enhance the adaptability and performance of the framework. Another important direction is to address the system and engineering challenges of real-world implementation, including developing lightweight models for edge deployment and robust communication protocols for large-scale AMoD fleets. By addressing these challenges, future work can move our framework closer to real-world application and help unlock its full potential in urban mobility systems.

IX. Acknowledgments

This work was supported by the Smart Traffic Fund (PSRI/29/2201/PR) of the Hong Kong SAR Government, China, and the General Research Fund (GRF) of the Research Grants Council of Hong Kong, China (HKU15209121; PolyU15207424).

References

[1] T. Chen, Z. Shen, S. Feng, L. Yang, and J. Ke, "Dynamic matching radius decision model for on-demand ride services: A deep multi-task learning approach," *Transportation Research Part E: Logistics and Transportation Review*, vol. 193, p. 103822, 2025.

[2] J. Gao, R. Cheng, Y. Wu, H. Zhao, W. Mai, and O. Cats, "Optimizing matching radius for ride-hailing systems with dual-replay-buffer deep reinforcement learning," *Computers & Industrial Engineering*, p. 111296, 2025.

[3] Z. Lei and S. V. Ukkusuri, "Scalable reinforcement learning approaches for dynamic pricing in ride-hailing systems," *Transportation Research Part B: Methodological*, vol. 178, p. 102848, 2023.

[4] J. Wang, S. Feng, and H. Yang, "Spatial–temporal upfront pricing under a mixed pooling and non-pooling market with reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[5] C. V. Beojone and N. Geroliminis, "Relocation incentives for ride-sourcing drivers with path-oriented revenue forecasting based on a markov chain model," *Transportation Research Part C: Emerging Technologies*, vol. 157, p. 104375, 2023.

[6] P. Zhu, I. I. Sirmatel, G. Ferrari-Trecate, and N. Geroliminis, "A coverage control-based idle vehicle rebalancing approach for autonomous mobility-on-demand systems," *IEEE Transactions on Control Systems Technology*, 2024.

[7] P. Zhu, G. Ferrari-Trecate, and N. Geroliminis, "Hierarchical control for vehicle repositioning in autonomous mobility-on-demand systems," *IEEE Transactions on Control Systems Technology*, 2024.

[8] T. Lyu, S. Feng, H. Liu, and H. Yang, "Llm-oddr: A large language model framework for joint order dispatching and driver repositioning," *arXiv preprint arXiv:2505.22695*, 2025.

[9] K. Xu, L. Sun, J. Liu, and H. Wang, "An empirical investigation of taxi driver response behavior to ride-hailing requests: A spatio-temporal perspective," *PloS one*, vol. 13, no. 6, p. e0198605, 2018.

[10] F. Zong, T. Wu, and H. Jia, "Taxi drivers' cruising patterns—insights from taxi gps traces," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 2, pp. 571–582, 2018.

[11] H. Chen, P. Sun, Q. Song, W. Wang, W. Wu, W. Zhang, G. Gao, and Y. Lyu, "i-rebalance: Personalized vehicle repositioning for supply demand balance," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 1, pp. 46–54, 2024.

[12] M. Han, P. Senellart, S. Bressan, and H. Wu, "Routing an autonomous taxi with reinforcement learning," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 2016, pp. 2421–2424.

[13] C. Mao, Y. Liu, and Z.-J. M. Shen, "Dispatch of autonomous vehicles for taxi services: A deep reinforcement learning approach," *Transportation Research Part C: Emerging Technologies*, vol. 115, p. 102626, 2020.

[14] Y. Jiao, X. Tang, Z. T. Qin, S. Li, F. Zhang, H. Zhu, and J. Ye, "Real-world ride-hailing vehicle repositioning using deep reinforcement learning," *Transportation Research Part C: Emerging Technologies*, vol. 130, p. 103289, 2021.

[15] N. Garg and S. Ranu, "Route recommendations for idle taxi drivers: Find me the shortest route to a customer!" in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1425–1434.

[16] M. M. Vazifeh, P. Santi, G. Resta, S. H. Strogatz, and C. Ratti, "Addressing the minimum fleet problem in on-demand urban mobility," *Nature*, vol. 557, no. 7706, pp. 534–538, 2018.

[17] P. Santi, G. Resta, M. Szell, S. Sobolevsky, S. H. Strogatz, and C. Ratti, "Quantifying the benefits of vehicle pooling with shareability networks," *Proceedings of the National Academy of Sciences*, vol. 111, no. 37, pp. 13 290–13 294, 2014.

[18] Y. Gao, D. Jiang, and Y. Xu, "Optimize taxi driving strategies based on reinforcement learning," *International Journal of Geographical Information Science*, vol. 32, no. 8, pp. 1677–1696, 2018.

[19] K. Lin, R. Zhao, Z. Xu, and J. Zhou, "Efficient large-scale fleet management via multi-agent deep reinforcement learning," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 1774–1783.
[20] Z. Xu, Z. Li, Q. Guan, D. Zhang, Q. Li, J. Nan, C. Liu, W. Bian, and J. Ye, "Large-scale order dispatch in on-demand ride-hailing platforms: A learning and planning approach," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 905–913.
[21] H. Rong, X. Zhou, C. Yang, Z. Shafiq, and A. Liu, "The rich and the poor: A markov decision process approach to optimizing taxi driver revenue efficiency," in *Proceedings of the 25th ACM international on conference on information and knowledge management*, 2016, pp. 2329–2334.
[22] J. Holler, R. Vuorio, Z. Qin, X. Tang, Y. Jiao, T. Jin, S. Singh, C. Wang, and J. Ye, "Deep reinforcement learning for multi-driver vehicle dispatching and repositioning problem," in *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019, pp. 1090–1095.
[23] J. Jin, M. Zhou, W. Zhang, M. Li, Z. Guo, Z. Qin, Y. Jiao, X. Tang, C. Wang, J. Wang *et al.*, "Coride: joint order dispatching and fleet management for multi-scale ride-hailing platforms," in *Proceedings of the 28th ACM international conference on information and knowledge management*, 2019, pp. 1983–1992.
[24] T. Verma, P. Varakantham, S. Kraus, and H. C. Lau, "Augmenting decisions of taxi drivers through reinforcement learning for improving revenues," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 27, 2017, pp. 409–417.
[25] Z. T. Qin, H. Zhu, and J. Ye, "Reinforcement learning for ridesharing: An extended survey," *Transportation Research Part C: Emerging Technologies*, vol. 144, p. 103852, 2022.
[26] H. Wei, Z. Yang, X. Liu, Z. Qin, X. Tang, and L. Ying, "A reinforcement learning and prediction-based lookahead policy for vehicle repositioning in online ride-hailing systems," *IEEE Transactions on Intelligent Transportation Systems*, 2023.
[27] J. Xie, Y. Liu, and N. Chen, "Two-sided deep reinforcement learning for dynamic mobility-on-demand management with mixed autonomy," *Transportation Science*, vol. 57, no. 4, pp. 1019–1046, 2023.
[28] S. Kim, M. E. Lewis, and C. C. White, "Optimal vehicle routing with real-time traffic information," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 2, pp. 178–188, 2005.
[29] S. Yu, J. Zhou, B. Li, S. Mabu, and K. Hirasawa, "Q value-based dynamic programming with sarsa learning for real time route guidance in large scale road networks," in *The 2012 international joint conference on neural networks (IJCNN)*. IEEE, 2012, pp. 1–7.
[30] J. Ke, F. Xiao, H. Yang, and J. Ye, "Learning to delay in ride-sourcing systems: A multi-agent deep reinforcement learning framework," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 5, pp. 2280–2292, 2020.
[31] C. V. Beojone, P. Zhu, I. I. Sirmatel, and N. Geroliminis, "A hierarchical control framework for vehicle repositioning in ride-hailing systems," *Transportation Research Part C: Emerging Technologies*, p. 104717, 2024.
[32] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.
[33] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving, "Fine-tuning language models from human preferences," *arXiv preprint arXiv:1909.08593*, 2019.
[34] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 1–43, 2012.
[35] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, no. 2-3, pp. 235–256, 2002.
[36] S. Feng, T. Chen, Y. Zhang, J. Ke, Z. Zheng, and H. Yang, "A multi-functional simulation platform for on-demand ride service operations," *Communications in Transportation Research*, vol. 4, p. 100141, 2024.
[37] F. S. Hillier and G. J. Lieberman, *Introduction to Operations Research*, 10th ed. McGraw-Hill, 2010.
[38] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.
[39] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1928–1937.
[40] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1026–1034, 2015.

X. Biography Section

Taijie Chen received the B.E. degree in software engineering from Nankai University, and an M.S. in Computer Science from the University of Hong Kong. Currently, he is a Ph.D. candidate in Civil Engineering at the University of Hong Kong. His research interests include machine learning and optimization in transportation systems.

Jingyun Liu holds a B.B.A. in Accounting and Finance from Hong Kong Baptist University and an M.S. in Computer Science from the University of Hong Kong. She is currently pursuing an M.S. in Software Engineering at Carnegie Mellon University, Silicon Valley. Her research interests focus on the application of deep learning in the smart traffic field.

Siyuan Feng received a B.S. degree in civil engineering from Tongji University, M.S. degree in Civil Engineering from University of California, Berkeley, and a Ph.D. in Civil and Environmental Engineering from Hong Kong University of Science and Technology. Currently, he is an Assistant Professor in the Department of Aeronautical and Aviation Engineering, Hong Kong Polytechnic University.

Jiandong Qiu is a Professor and Senior Engineer with the Department of Transportation Information and Modeling at SUTPC. He is recognized as a distinguished member of Shenzhen's high-level talent program. Dr. Qiu has over 17 years of experience in traffic modeling, simulation, and big data analytics. He has led more than 30 major urban transportation projects and directed 10 national and provincial research initiatives.

Jintao Ke received a B.S. degree in civil engineering from Zhejiang University and Ph.D. degree in Civil and Environmental Engineering from Hong Kong University of Science and Technology. He is now an Assistant Professor (AP) in the Department of Civil Engineering, the University of Hong Kong. His research interests include spatio-temporal traffic forecasting, shared mobility (ride-sourcing and ride-sharing services), transportation economy, machine learning in transportation, urban computing.

APPENDIX

In this study, we employed the same simulation framework as described in our previous work. For detailed information regarding the simulator's architecture and functionalities, please refer to the sections outlined below.

1. Input Data and Agent Properties

The input data comprises historical order distributions with spatio-temporal details on origin-destination (OD) pairs and geographic information about the Manhattan road network, with 4,474 nodes and 9,682 edges in the study area. These datasets are essential for modeling realistic traffic conditions and vehicle-passenger interactions within the simulated environment.

2. Order Dispatching and Response After Assignment

Order dispatching is formulated as a bipartite matching challenge, addressed through an Integer Linear Programming (ILP) formulation ([37]). The Kuhn-Munkres (KM) algorithm ([38]) is employed to solve the ILP efficiently, optimizing the matching process between vehicles and passengers.

Following the assignment, both passengers and vehicles make decisions regarding their matched counterparts. Matched passengers can cancel their orders if the estimated pickup time is excessively long. Similarly, unmatched passengers may cancel their orders if the time required to find a responsible vehicle becomes untenable.

This simulator setup allows for the detailed exploration and evaluation of response strategies in dynamic ride-sharing scenarios, underpinning the model's potential applications in smart traffic systems.

3. Order Generation and Pricing

During each simulated time interval, new orders are generated reflective of the predefined origin-destination (OD) demand pattern. This process involves bootstrapping from an existing pool of historical ride-sourcing data, which serves as a realistic basis for simulation scenarios. Concurrently, a pricing module is integrated within the order generation framework. As new ride-sourcing orders materialize in the simulated market, their prices are automatically determined based on the established taxi trip fare structure outlined by the Taxi and Limousine Commission of New York City. This integration ensures that the simulated pricing dynamics mirror real-world conditions.

4. Idle Vehicle Repositioning

Idle vehicle repositioning represents a critical component of our simulation, where the Top to Bottom (T2B) algorithm is operationalized. Should a vehicle remain idle in a specific region for an extended duration, the T2B algorithm is triggered, directing the vehicle either to remain in the current region or to relocate based on current spatio-temporal data. For each designated target region, a pre-planned route is generated. The vehicle then follows this route, adhering to the trajectory provided by the T2B algorithm, to navigate towards and within the target region. This procedure not only optimizes the distribution of idle vehicles but also enhances the efficiency of the overall transportation network within the simulated environment.

5. Routing

The open-source road network tool, OSMNX, is employed to facilitate route planning through the road network. Utilizing this API, the simulator retrieves the shortest route for any given origin-destination (OD) pair, leveraging geographical data from the OpenStreetMap network. To expedite the routing process, the shortest paths are persisted in MongoDB, enhancing the efficiency and response time of the routing module.

Similarly, passengers are subject to a waiting time threshold, which includes both the order match time and the pickup time. Should this combined waiting period exceed a predetermined limit, the passenger's order will be automatically canceled, and the passenger will be removed from the simulation. This mechanism ensures that the simulation reflects realistic user behavior, where prolonged wait times typically lead to cancellations.

We implement an Advantage Actor-Critic (A2C) model ([39]), consisting of separate neural networks for the actor and the critic. Both networks are structured with fully connected layers and employ the ReLU activation function for hidden layers. The actor uses a softmax output layer to provide a policy distribution over actions, while the critic outputs a scalar value estimate using a linear activation function.

Model Parameters

The model operates with a discount factor of 0.95. The learning rates for the actor and critic are set to 0.001 and 0.005, respectively. Below, we detail the architecture and parameters of each network component:

Both models are implemented using the sequential model architecture in a deep learning framework. Each model layer is initialized with 'he_uniform'([40]) for robust convergence during training. The actor model compiles with a categorical cross-entropy loss to align with the policy output, whereas

TABLE V: Architecture of the actor network

Layer (type)	Output shape	Activation	Param #
Dense	32	ReLU	832
Dense	64	ReLU	2112
Dense	35	Softmax	2275

TABLE VI: Architecture of the critic network

Layer (type)	Output shape	Activation	Param #
Dense	32	ReLU	832
Dense	64	ReLU	2112
Dense	1	Linear	65

the critic uses mean squared error loss reflecting the value estimation task.