Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 9402671, 14 pages
https://doi.org/10.1155/2022/9402671

Research Article

Minimizing the Cost of Spatiotemporal Searches Based on Reinforcement Learning with Probabilistic States

Lei Han[1], **Chunyu Tu**[2], **Zhiyong Yu**[2], **Fangwan Huang**[2], **Wenzhong Guo**[2], **Chao Chen**[3] and **Zhiwen Yu**[1]

[1]*School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China*
[2]*College of Mathematics and Computer Sciences, Fuzhou University, Key Laboratory of Spatial Data Mining and Information Sharing, Ministry of Education, and Fujian Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou 350108, China*
[3]*School of Computer Science, Chongqing University, Chongqing 400044, China*

Correspondence should be addressed to Fangwan Huang; hfw@fzu.edu.cn

Received 2 March 2022; Accepted 29 April 2022; Published 8 June 2022

Academic Editor: Han Liu

Portraying the trajectories of certain vehicles effectively is of great significance for urban public safety. Specially, we aim to determine the location of a vehicle at a specific past moment. In some situations, the waypoints of the vehicle's trajectory are not directly available, but the vehicle's image may be contained in massive camera video records. Since these records are only indexed by location and moment, rather than by contents such as license plate numbers, finding the vehicle from these records is a time-consuming task. To minimize the cost of spatiotemporal search (a spatiotemporal search means the effort to check whether the vehicle appears at a specified location at a specified moment), this paper proposes a reinforcement learning algorithm called Quasi-Dynamic Programming (QDP), which is an improved Q-learning. QDP selects the searching moment iteratively based on known past locations, considering both the cost efficiency of the current action and its potential impact on subsequent actions. Unlike traditional Q-learning, QDP has probabilistic states during training. To address the problem of probabilistic states, we make the following contributions: 1) replaces the next state by multiple states of a probability distribution; 2) estimates the expected cost of subsequent actions to calculate the value function; 3) creates a state and an action randomly in each loop to train the value function progressively. Finally, experiments are conducted using real-world vehicle trajectories, and the results show that the proposed QDP is superior to the previous greedy-based algorithms and other baselines.

1. Introduction

Portraying the trajectories of some vehicles in a city is important to foresee potential safety issues, for example, to infer the intention of a suspicious vehicle. We only get the locations of the vehicle at some key moments, instead of at all moments for economic efficiency. Figure 1 shows a typical application scenario. The sequential numbers indicate the waypoints of the vehicle trajectory from 8 : 00 to 9 : 00 on a certain day, of which the yellow ones are the locations of some key moments. By locating a suspicious vehicle at certain times (such as target time, etc.) and marking them on a map until we can get a rough idea of its trajectory.

To track vehicles in a city, vehicle Re-Identification (Re-Id) [1–4] focuses on how to identify the target vehicle in camera video records as accurately as possible. However, Re-Id does not consider how to select records, which are massive and only indexed by location and moment. As a result, Re-Id needs to be performed on all records exhaustively. With the rapid urbanization, there are a large number of cameras in the city. By searching within the video records of these cameras, it is possible but time-consuming to find the target vehicle at the target moment, because these records are massive and only indexed by location and moment, rather than by the contents such as the license plate number. Besides the straightforward strategy that all searches are performed at

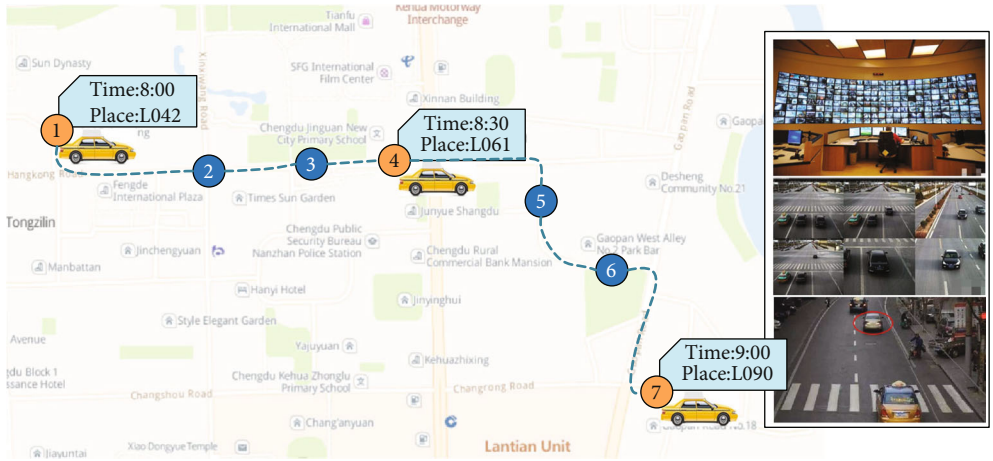


Figure 1: Vehicle location at specific moments.

the target moment, another strategy worth exploring is to spend some searches at intermediate moments to determine the vehicle's likely subsequent path, thus significantly narrowing the search area at the target moment. We call the effort to check whether the vehicle appears at a specified location at a specified moment as a spatiotemporal search.

How to determine the location of a vehicle at the target moment with the minimumcost of spatiotemporal searches in massive camera video records? The cost of a spatiotemporal search can be considered as a constant for simplicity, regarding either human or artificial intelligence. In previous work [5], we proposed two greedy algorithms named Intermediate Searching at Heuristic Spatiotemporal Units (IHUs) and Intermediate Searching at Heuristic Moments (IHMs), and some other simple but uncompetitive baselines. Compared with the algorithm IHUs that considers both moments and location, IHMs can fully utilize the information gained from previous failed searches, performing location-by-location searches at the same time. Since IHMs is based on the greedy heuristic, it may result a locally optimal solutions. The most common way to go beyond the locally optimal solution is dynamic programming. Dynamic programming is essentially an exhaustive idea. By comparing all solutions, it can guarantee that the solution is globally optimal. However, there are too many spatiotemporal points to be considered, making the solution space too large to list all solutions. Reinforcement learning can alleviate the problem through quasi-dynamic programming process. The spatiotemporal search strategy based on reinforcement learning selects an intermediate moment according to known vehicle spatiotemporal points, taking into account the cost efficiency of the current action and its potential impact on subsequent actions. The strategy of spatiotemporal search is a standard finite Markov decision process [6]. Q-learning, a reinforcement learning algorithm, has been proved to eventually converge to the optimum action values with probability 1 so long as all actions are repeatedly sampled in all states and the action values are represented discretely [7, 8]. However, if adopting Q-learning directly to address our problem, since the next state (known spatiotemporal point of the vehicle trajectory) cannot be determined after the action (i.e., selecting an intermediate moment and performing the corresponding spatiotemporal searches), it is difficult to update the next state and the value function when training. To overcome this challenge, this paper proposes Quasi-Dynamic Programming (QDP) algorithm to improve the training phase of Q-learning. The contributions of this paper include:

(1) We propose a reinforcement learning algorithm called QDP that can result the near optimal solution to the problem of minimizing spatiotemporal search cost

(2) To address the challenge of probabilistic state in the training phase, we propose a novel training method for QDP, which replaces the next state by multiple states with a probability distribution, estimates the expected search cost of subsequent selections to calculate the value function, and creates a state and an action randomly in each loop to train the value function progressively

(3) We evaluate the proposed QDP on real-world vehicle trajectories, and the results show that it is superior to the previous greedy-based algorithms (IHMs) and other baselines

The contents of this paper are arranged as following: Section 2 discusses some related work; Section 3 proposes the overall process of QDP, and explains the training method for QDP; then the experiments are conducted in Section 4; finally, the conclusions and future work are summarized in Section 5.

2. Related Work

Since this paper addresses the search problem based on reinforcement learning, we will introduce the related work from the two aspects of search theory and reinforcement learning.

2.1. Search Theory. The core issue of search theory is designing a search strategy to find a missing target, under constraints such as time limit, forces required, and task

budget. Related work mainly comes from the field of operations research or cybernetics.Koopman [9–11] used the basic probability theory to optimize the conFigureuration of budgeted search forces to find stationary targets. Stone [12, 13] presented detailed mathematical assumptions and theoretical proofs for specific problems of moving target searching and found the necessary conditions for the optimal search strategy. Besides the above theoretical analysis, some other works adopted the search theory in practical application scenarios. Researchers [14–17] established a management framework of task planning and feedback controlling based on the classical search theory. For the task planning of satellite SPOT-5, Bensana [18, 19] and Agnese [20] compared the computational performance of complete searching algorithms (i.e., Depth-first Search, Dynamic Programming, Russian Doll Search) and incomplete searching algorithms (i.e., Greedy Search, Tabu Search) under different problem scales. To locate the target to be observed in a region by satellite scanning, Lemaitre et al. [21] discussed Greedy Search, Dynamic Programming, Constrained Programming, and Genetic Algorithms to solve the problem. Different from the existing work, we study a more realistic problem with high rate of knowledge update in the application scenario of vehicle tracking.

2.2. Reinforcement Learning. Reinforcement learning [22, 23] is used to describe and solve the problem that the agent learns to obtain the maximum reward or achieve a specific goal in the process of interacting with the environment. Reinforcement learning is widely used in areas such as process control, task scheduling, robotics, and smart games [24–26]. Some complex reinforcement learning algorithms can reach or even surpass the human level in certain tasks [27, 28]. Q-learning is one reinforcement learning algorithm we pay attention to in this paper. Watkins [29] proposed Q-learning first, which utilizes the reward of state-action pair for value function updating. Rummery and Niranjan [30] proposed SARSA, a reinforcement learning algorithm with online policy based on Q-learning. Harm van Seijen et al. [31] presents a theoretical and empirical analysis of Expected Sarsa, a variation on Sarsa, the classic on-policy temporal difference method for model-free reinforcement learning. DeepMind [32] proposed DQN (Deep Q-Networks) by combining convolutional neural networks and Q-learning. Different from traditional Q-learning that the next state is uniquely determined, in this paper, the next state is multiple states of a probability distribution, i.e., probabilistic states. We propose a novel training method to overcome the difficulty brought by probabilistic states.

3. Overall Process of Quasi-Dynamic Programming (QDP)

Symbols used in this paper are shown in Notations. Let $O = \{o_1, o_2, \cdots, o_{|O|}\}$ be vehicles, $L = \{l_1, l_2, \cdots, l_{|L|}\}$ be locations of a city, $D = \{d_1, d_2, \cdots, d_{|D|}\}$ be days, $Tr = \{tr_1, tr_2, \cdots, tr_{|Tr|}\}$ be trajectories. In this paper, a trajectory is defined as a sequence of spatiotemporal points, i.e., $tr(o_x, d_j) = \langle (t_1, l_{t_1}), (t_2, l_{t_2}), \cdots \rangle | o_x, d_j$.

Definition 1 (a spatiotemporal search). The effort to check whether the target vehicle o_x appears at a specified location l_i at a specified moment t_{opt} in the camera records of that spatiotemporal point (t_{opt}, l_i). If vehicle o_x is found at location l_i at moment t_{opt}, ie $(t_{opt}, l_i) \in tr(o_x, d_j)$, it will returns 1, otherwise it returns 0. Formally,

$$s(o_x, t_{opt}, l_i) = \begin{cases} 1, & (t_{opt}, l_i) \in tr(o_x, d_j) \\ 0, & else \end{cases} \tag{1}$$

Problem 2 (tracking an object by spatiotemporal search). Give an object o_x, a day d_x, a past moment t_p of that day, o_x's location l_{t_p} at that moment, a current moment t_x, camera records that make the spatiotemporal search $s(o_x, t_k, l_i)$ available in d_x before/at t_x, as well as history trajectories TR', in which arbitrary $d_j < d_x$. When and where to execute a spatiotemporal search can base on TR' and the outputs of previous searches. The problem is how to utilize a minimal number of searchs to find the object at current moment t_x. Formally,

$$\begin{aligned} &\textbf{give } o_x, d_x, t_x, \left(t_p, l_{t_p}\right), TR' \text{ before } s(o_x, t_k, l_i) \text{ in } d_x \\ &\quad \textbf{return } l_x, \text{ s.t. } \min_{s(o_x, t_x, l_x)} |\langle s(), s(), \cdots, s(o_x, t_x, l_x)\rangle| \end{aligned} \tag{2}$$

The difficulty of a spatiotemporal search is affected by many factors in the spatiotemporal point (e.g., traffic flow, road network, and building density). It needs to be quantified by professionals or artificial intelligence. Let $C = \{c_{l_1}^1, c_{l_2}^1, \cdots, c_{l_{|L|}}^1, \cdots, c_{l_1}^{|d_x|}, c_{l_2}^{|d_x|}, \cdots, c_{l_{|L|}}^{|d_x|}\}$ contains the search cost per spatiotemporal point on day d_x. It can be seen from the problem definition that we have to find the vehicle finally, so we use the cost of spatiotemporal search as the only indicator to measure the performance of different algorithms.

The overall process of QDP is shown in Figure 2. The process includes two phases: the training phase and executing phase. The training phase outputs the location decision model and the moment decision model. The location decision model is trained with the help of mobility prediction and basic optimal searching. This model will return the optimal searching location sequence at a given moment. The moment decision model is trained based on Monte-Carlo method and probabilistic states. This model will return the optimal moment that will be searched in the next action. The executing phase describes the online spatiotemporal search utilizing the trained moment decision model and location decision model.

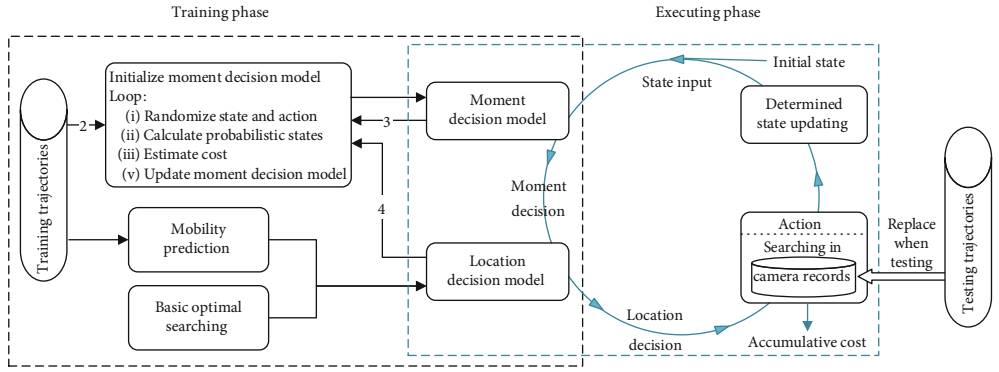


Figure 2: Overall Process of Quasi-Dynamic Programming (QDP).

3.1. Training the Location Decision Model. The location decision model is responsible to output the location decision $\langle t_{opt}, \xi_{opt} \rangle$ (the optimal searching location sequence at moment t_{opt} given by the moment decision model). It is trained with the help of mobility prediction and basic optimal searching. The mobility prediction algorithm is not the focus of this paper. In the previous work [5], we have analyzed the advantages of choosing the first-order Markov model for mobility prediction. Suppose that the transition probability matrix $TPM_{\Delta t}^{|L|\times|L|}$ represents the probabilities of a vehicle moving from one location to another after a period of time $\Delta t = t_{opt} - t_s$, of which an element is $p_{l_i}^{opt}$, i.e., the probability of vehicle o_x moves from location l_s at moment t_s to location l_i at moment t_{opt}.

$$TPM_{\Delta t}^{|L|\times|L|} = \left\{ p_{l_i}^{opt} \right\} \tag{3}$$

$$p_{l_i}^{opt} = \frac{\#tr()with(t_s, l_s)\&(t_{opt}, l_i)}{\#tr()with(t_s, l_s)}, t_s = t_1, t_2, \cdots, tr() \in TR' \tag{4}$$

where the meaning of "#" is "the number of", $tr()$ is a trajectory. Equation Equation (4) represents the ratio of "the number of trajectories that pass both (t_s, l_s) and (t_{opt}, l_i)" to "the number of trajectories of only the pass (t_s, l_s)". $TPM_{\Delta t}^{|L|\times|L|}$ can be trained from historical vehicle trajectories. When given an original location, a row of probabilities is output as the predicted result of the destination location.

The basic optimal searching theory can assign the priority of locations to search at. Suppose the searching location sequence at moment t_{opt} is $\langle t_{opt}, \xi_{opt} \rangle = \langle (t_{opt}, l_1^{\xi_{opt}}), (t_{opt}, l_2^{\xi_{opt}}), \cdots, (t_{opt}, l_i^{\xi_{opt}}), \cdots \rangle, l_i^{\xi_{opt}} \in L$.

Definition 3 (probability-cost ratio). The ratio of the probability $p_{l_i^{\xi}}^{opt}$ of finding the target vehicle to the cost $c_{l_i^{\xi}}^{opt}$ of the $i-th$ spatiotemporal search in the sequence $\langle t_{opt}, \xi_{opt} \rangle$, formally,

$$\eta\left(t_{opt}, l_i^{\xi_{opt}}\right) = \frac{p_{l_i^{\xi}}^{opt}}{c_{l_i^{\xi}}^{opt}} \tag{5}$$

Theorem 4. *[33]: The necessary and sufficient condition for the sequence $\langle t_{opt}, \xi_{opt} \rangle$ to be the searching location sequence with the lowest search cost at moment t_{opt} is that the probability-cost ratios are in descending order.*

Consequently, the location decision can be made according to Theorem 4.

We need to estimate the search cost which will be used during training the moment decision model. The optimal sequence $\langle t_{opt}, \xi_{opt} \rangle$ has a expected search cost:

$$Cost\left(t_{opt}, \xi_{opt}\right) = \sum_{i=1}^{|L|} p_{l_i^{\xi}}^{opt} \times \sum_{j=1}^{i} c_{l_j^{\xi}}^{opt} \tag{6}$$

To demonstrate how to calculate the expected search cost using Equation (6), we assume the spatiotemporal search costs $c_{l_1}^3 = c_{l_2}^3 = c_{l_3}^3 = c_{l_4}^3 = 1$, the location transition probabilities $p_{l_1}^3 = 1/3, p_{l_2}^3 = 0, p_{l_3}^3 = 2/3, p_{l_4}^3 = 0$. Then, the descendingly sorted location sequence is $\langle t_3, \xi_3 \rangle = \langle (t_3, l_3), (t_3, l_1), (t_3, l_2), (t_3, l_4) \rangle$ at moment t_3. Then the expected search cost is:

$$\begin{aligned} Cost(t_3, \xi_3) &= p_{l_3}^3 \times c_{l_3}^3 + p_{l_1}^3 \times \left(c_{l_3}^3 + c_{l_1}^3\right) + p_{l_2}^3 \times \left(c_{l_3}^3 + c_{l_1}^3 + c_{l_2}^3\right) \\ &\quad + p_{l_4}^3 \times \left(c_{l_3}^3 + c_{l_1}^3 + c_{l_2}^3 + c_{l_4}^3\right) = \frac{2}{3} \times 1 + \frac{1}{3} \\ &\quad \times (1+1) + 0 \times (1+1+1) + 0 \times (1+1+1+1) \\ &= \frac{4}{3} \end{aligned} \tag{7}$$

3.2. Training the Moment Decision Model. Perform spatiotemporal searches at moment t_{opt} should consider both the cost-efficiency of the current action and its potential impact on subsequent actions. So, how to select the next searching moment t_{opt} when knowing a spatiotemporal point (t_s, l_s)

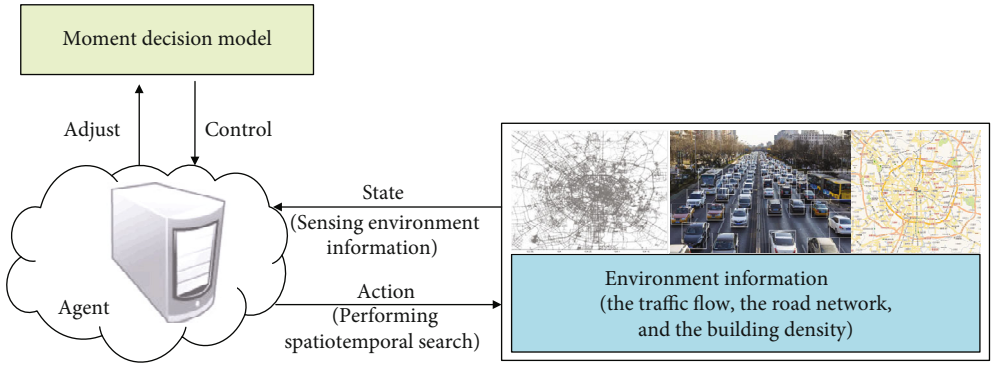


Figure 3: Training of the moment decision model.

of target vehicle o_x? The moment decision model is responsible to answer this question.

Following reinforcement learning, the moment decision model is represented as (S, A, C, Γ, Q), in which $S_{l_s}^s \in S$ denotes the current state (t_s, l_s); $A_{l_s}^s \in A$ denotes the current action $\langle t_{opt}, \xi_{opt} \rangle$; $C : S \times A \longrightarrow C$ denotes the search cost function $C^{opt} = Cost(t_{opt}, \xi_{opt})$; $\Gamma : S \times A \longrightarrow \Delta S$ denotes the state transition function; Q denotes the value function. The next state $S_{l_s}^{s\prime}$ is obtained by performing action $A_{l_s}^s$ at the state $S_{l_s}^s$, and the value function is updated using Equation (8), where α is the learning rate between (0, 1], and γ is the discount rate.

$$Q\left(S_{l_s}^s, A_{l_s}^s\right) = Q\left(S_{l_s}^s, A_{l_s}^s\right) + \alpha.\left(C^{opt} + \gamma.\min_{A_{l_s}^{s\prime}} Q\left(S_{l_s}^{s\prime}, A_{l_s}^{s\prime}\right) - Q\left(S_{l_s}^s, A_{l_s}^s\right)\right) \tag{8}$$

The learning rate α can control the update rate of the value function. The discount rate γ can balance the importance between the immediate and potential reward. We use $Q(S_{l_s}^{s\prime}, A_{l_s}^{s\prime})$to estimate the long-term impact of performing action $A_{l_s}^s$ at state $S_{l_s}^s$.

Figure 3 shows how to train the moment decision model. First of all, the moment decision model is initialized. In each loop, the moment decision model decides a searching moment based on the current state, and the location decision model decides the corresponding searching location sequence. Then the value function is adjusted according to the potential impact on subsequent actions and the sensed environmental information (e.g., traffic flow, road network, and building density, which will be quantified as search cost C^{opt}). The moment decision model will continually repeat the above loop until the determined searching moment t_{opt} at the current state $S_{l_s}^s$ is optimal.

To train the moment decision model in QDP, this paper proposes a training method based on Monte-Carlo method and probabilistic states to solve the problem, as shown in Figure 4.

In the training of QDP, performing the action $A_{l_s}^s = \langle t_{opt}, \xi_{opt} \rangle$ at the state $S_{l_s}^s = (t_s, l_s)$ will result in a probability distribution $\{p_{l_1^\xi}^{opt}, p_{l_2^\xi}^{opt}, \cdots, p_{l_{|L|}^\xi}^{opt}\}$ at different locations $\{l_1^{\xi_{opt}}, l_2^{\xi_{opt}}, \cdots, l_{|L|}^{\xi_{opt}}\}$ at moment t_{opt}. We use Equation (9) to estimate the long-term impact of performing action $A_{l_s}^s$ at state $S_{l_s}^s$, i.e., the minimum cost of subsequent searches. As shown in Figure 5, $\{S_{l_i^\xi}^{opt}\}$ are the next states that can be generated by performing action $A_{l_s}^s$ (i.e., spatiotemporal searches) at state $S_{l_s}^s$, and $\{A_{l_1^\xi}^{opt}\}$, $\{A_{l_2^\xi}^{opt}\}$, ... are actions that can be performed at state $S_{l_1^\xi}^{opt}$, $S_{l_2^\xi}^{opt}$, ..., respectively. Especially, when the agent finds the vehicle's location at the target moment $t_{t\ \arg et}$, it no longer needs to perform any action or update the state.

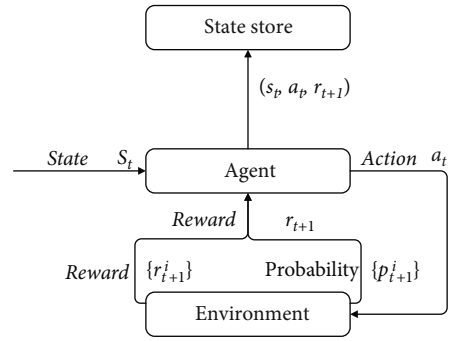


Figure 4: Training of QDP. $s_t$denotes the current state, a_t denotes the action performed at state s_t, r_{t+1} denotes the reward (reward is the inverse of cost) of the action a_t at state s_t, $\{p_{t+1}^i\}$ denotes the probability distribution of next states, and r_{t+1}^i denotes the reward of the optimal action at each possible state.

$$\min_{A_{l_s}^{s\prime}} Q\left(S_{l_s}^{s\prime}, A_{l_s}^{s\prime}\right) = \sum_{i=1}^{|L|} p_{l_i^\xi}^{opt} \times \min_{A_{l_i^\xi}^{opt}} Q\left(S_{l_i^\xi}^{opt}, A_{l_i^\xi}^{opt}\right) \tag{9}$$

To solve the problem that QDP is difficult to update the state during training, we adopt the idea of Monte-Carlo method. In each loop, we randomize the current state and action, calculate the expected cost of the subsequent searches by Equation (9) for the value function updated by Equation (8), and iterate the loop until the

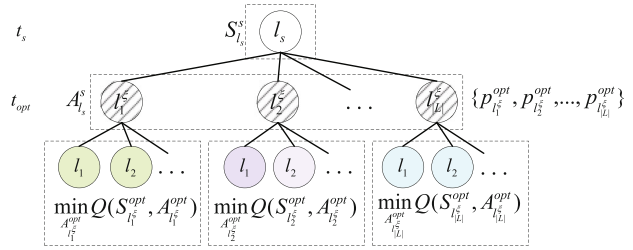


Figure 5: Estimating the subsequent search cost with probabilistic states.

value function converges. The pseudo code of QDP's training method is shown as Algorithm 1.

3.3. Spatiotemporal Searches at Executing Phase. The basic idea of QDP's executing phase is greedy. At each step, the moment decision model decides the optimal searching moment t_{opt} based on known spatiotemporal point (t_s, l_s) of the vehicle o_x, and the location decision model decides the corresponding searching location sequence $\langle t_{opt}, \xi_{opt}\rangle$. We perform spatiotemporal searches in camera records to find the vehicle o_x's location $l_i^{\xi_{opt}}$ at the moment t_{opt}, and update the known spatiotemporal point $(t_s, l_s) = (t_{opt}, l_i^{\xi_{opt}})$. The above steps are iterated until the vehicle o_x's location $l_{t\ \mathrm{arg}\ et}$ at the target moment $t_{t\ \mathrm{arg}\ et}$ is found. Finally, QDP outputs the accumulative search cost, i.e., the cost of spatiotemporal search. The complete steps of spatiotemporal search at QDP's executing phase are as follows:

(1) Quantify the search cost $C = \{c_{l_1}^1, c_{l_2}^1, \cdots, c_{l_{|L|}}^1, \cdots, c_{l_1}^{|d_x|}, c_{l_2}^{|d_x|}, \cdots, c_{l_{|L|}}^{|d_x|}\}$ at different moments and at different locations of the day d_x

(2) Initialize the known spatiotemporal point (t_s, l_s) of the target vehicle o_x and the target moment $t_{t\ \mathrm{arg}\ et}$ in the testing day d_x

(3) Decide the optimal searching moment $t_{opt}, t_{opt} \in (t_s, t_{t\ \mathrm{arg}\ et}]$, according to the moment decision model

(4) Decide the searching location sequence $\langle t_{opt}, \xi_{opt}\rangle$ according to the location decision model

(5) Perform spatiotemporal searches $\langle (s(o_x, t_{opt}, l_1^{\xi_{opt}})), (s(o_x, t_{opt}, l_2^{\xi_{opt}})), \cdots\rangle$ until $s(o_x, t_{opt}, l_i^{\xi_{opt}}) = 1$, then return the vehicle o_x's location $l_i^{\xi_{opt}}$ at moment t_{opt}

(6) Update known spatiotemporal point $(t_s, l_s) = (t_{opt}, l_i^{\xi_{opt}})$

Repeat steps (3), (4), (5), and (6) until $t_s = t_{t\ \mathrm{arg}\ et}$, output the vehicle o_x's location $l_{t\ \mathrm{arg}\ et}$ at the target moment $t_{t\ \mathrm{arg}\ et}$ and the accumulative cost.

4. Evaluation

4.1. Dataset. The experimental dataset (The experimental dataset of this article is provided on demand, please contact the author if necessary) is derived from the trajectories of 19,000 taxis in Chengdu, China, in August 2014. The data acquisition area is about 30KM × 30KM, and the duration of each trajectory is from 7 : 00 am to 21 : 59 pm.

The GPS waypoints of different vehicles in the original dataset are unordered. In addition, some vehicles contain various numbers of waypoints within 1 minute while some contain no waypoints. Therefore, it is firstly necessary to preprocess the dataset. In this experiment, we generate a GPS waypoint sequence of a taxi by filtering the vehicle ID and sorting the timestamps. The time is discretized into moments, each with 1 minute, and only the waypoint with the earliest timestamp per minute indicates the location of the vehicle at that moment. Trajectories with seriously missing data are discarded. The raw data details and a complete trajectory are shown in Table 1.

The data acquisition area is divided into grids each of size 1KM × 1KM, and the GPS waypoints are projected into the grids, and each grid represents a location. The gird size of 1.2KM × 1.2KM and 1.5KM × 1.5KMare also tested in our experiments. Smaller grids are not appropriate since there is enough historical data in a smaller grid. The 99,265 trajectories of the first 14 days are used for training, and randomly selected 1000 trajectories from 21,963 trajectories in the last 3 days are used for testing, i.e., 1000 tests. In this experiment, it is assumed that the search cost C at a spatiotemporal point is 1. Finally, under different grid sizes, we perform 1000 tests and average the results.

4.2. Parameters. Two parameters of QDP need to be tuned: the learning rate α and the discount rate γ in Equation (8). We choose them from practical experiences.

The learning rate can control the update rate of the value function. Small learning rate will reduce the convergence speed of the value function, and a large learning rate may fail to converge to the optimal solution. To find a proper learning rate, we compare the test results (i.e., the accumulative search cost) of QDP with different learning rates ($\alpha = 1$, 0.5, 0.3, 0.1, respectively) under the same other settings (i.e., grid size = 1KM × 1KM, $t_{t\ \mathrm{arg}\ et} - t_s = 30$ min, discount rate $\gamma = 1$, and start moment ts =8 : 00/10 : 00/12 : 00/14 : 00/16 : 00/18 : 00/20 : 00).

The test results verify that the value function is converged with all different learning rates. As shown in Table 2, QDP with different α output the same accumulative cost, which implies their value functions are converged to the same solution. Only the training time is different. The training is performed on a computer with a memory of 16 GB and a processor of Inter(R) Core(TM) i7-6700HQ. To speed up the training, we choose $\alpha = 1$.

Discount rate γ can balance the importance between the immediate and potential reward. $\gamma > 1$ indicates that the immediate reward is more important than the potential reward, and vice versa. To find a proper discount rate, we compare the test results of QDP with different discount rates

Input: $t_{t\ arg\ et}$, C, α, γ, $TR'\ before\ d_x$
Output: value function Q
1 Calculate the vehicle transition probability $\{P(t_s, l_s, t_{opt}, :)\}$ at $\{(t_s, l_s)\}$ and at the moment $\{t_{opt}\}, t_{opt} \in (t_s, t_{t\ arg\ et}]$ according to TR'; $//P(t_s, l_s, t_{opt}, :) = \{p_{l_1}^{t_{opt}}, p_{l_2}^{t_{opt}}, \cdots, p_{l_{|L|}}^{t_{opt}}\} = TPM$
2 Calculate the expected cost $\{C(t_s, l_s, t_{opt})\}$ of spatiotemporal searches according to Equation (6); $//C(t_s, l_s, t_{opt}) = C^{opt} = Cost(t_{opt}, \xi_{opt})$
3 Initialization value function Q;
4 **while** Q does not converge:
5 Randomize spatiotemporal point (t_s, l_s) and the moment decision $t_{opt}, t_{opt} \in (t_s, t_{t\ arg\ et}]$;
6 Calculate $\min_{A_{l_s}^{s\prime}} Q(S_{l_s}^{s\prime}, A_{l_s}^{s\prime})$ according to Equation (9);
7 Update the value function $Q((t_s, l_s), t_{opt})$ according to Equation (8);
8 **end.**

Algorithm 1: QDP's training method

Table 1: Detailed description of the dataset.

(a) Raw data details

Field name	Example	Remarks
Vehicle ID	1	\
Passenger or not	0/1	\
Timestamp	1501584540	Unix timestamp, in seconds
Longitude	104.042833	GCJ-02
Latitude	30.599851	GCJ-02

(b) Complete trajectory of a vehicle in a day

Vehicle	Day	Longitude	Latitude	Moment
o_x	2014/8/4	104.039163	30.597572	7 : 00
		104.126565	30.599733	7 : 01
		...	...	...
		104.042833	30.599851	21 : 58
		104.127154	30.600009	21 : 59

(γ=0.97, 0.99, 1, 1.01, 1.03, 1.05, respectively) under the same other settings (i.e., grid size = 1KM × 1KM, $t_{t\ arg\ et} - t_s = 30$ min, learning rate $\alpha = 1$, and start moment ts traverses every moment from 7 : 00 to 21 : 29 of a day).

As shown in Table 3, when the discount rate $\gamma = 1$, the search cost of QDP is the smallest, so we set the discount rate γ to 1.

4.3. Baselines. In this section, we will describe three spatiotemporal searching algorithms as baselines: ALT, IEM, and IHMs [5]. Given the location l_s of the vehicle o_x at moment t_s of d_j, the process of the three algorithms are as follows:

4.3.1. All Searching at the Last Time (ALT). Calculate the probability $(p_{l_1}^{t\ arg\ et}, p_{l_2}^{t\ arg\ et}, \cdots)$ that the vehicle o_x moves to different locations at moment $t_{t\ arg\ et}$ according to the spatiotemporal point (t_s, l_s) and the historical trajectories TR'; Determine the searching location sequence $\langle t_{t\ arg\ et}, \xi_{t\ arg\ et}\rangle$ according to the location decision model; Perform spatiotemporal searches $\langle (s(o_x, t_{t\ arg\ et}, l_1^{\xi_{t\ arg\ et}})), (s(o_x, t_{t\ arg\ et}, l_2^{\xi_{t\ arg\ et}})), \cdots\rangle$ until $s(o_x, t_{t\ arg\ et}, l_i^{\xi_{t\ arg\ et}}) = 1$, then return the location $l_{t\ arg\ et} = l_i^{\xi_{t\ arg\ et}}$ of vehicle o_x at moment $t_{t\ arg\ et}$.

4.3.2. Intermediate Searching at an Estimated Moment (IEM). Calculate probability $(p_{l_1}^{opt}, p_{l_2}^{opt}, \cdots)$ that the vehicle o_x moves to different locations at moment $t_{opt}, t_{opt} \in (t_s, t_{t\ arg\ et})$ according to the spatiotemporal point (t_s, l_s) and the historical trajectories TR'; Determine the searching location sequence $\langle t_{opt}, \xi_{opt}\rangle$ according to the location decision model; Calculate probability $\{(p_{l_1}^{t\ arg\ et}, p_{l_2}^{t\ arg\ et}, \cdots)\}$ that the vehicle o_x moves to different locations at moment $t_{t\ arg\ et}$ according to the spatiotemporal point $\{(t_{opt}, l_i^{\xi_{opt}})\}$ and TR'; Determine the searching location sequence $\{\langle t_{t\ arg\ et}, \xi_{t\ arg\ et}\rangle\}$. The above two steps are only used for cost estimation of different t_{opt}, but no need to perform the searches actually. Determine an intermediate search moment t_{opt} according to Equation (10); Perform spatiotemporal searches $\langle (s(o_x, t_{opt}, l_1^{\xi_{opt}})), (s(o_x, t_{opt}, l_2^{\xi_{opt}})), \cdots\rangle$, until $s(o_x, t_{opt}, l_i^{\xi_{opt}}) = 1$, update to spatiotemporal point $(t_{opt}, l_i^{\xi_{opt}})$; Perform spatiotemporal searches $\langle (s(o_x, t_{t\ arg\ et}, l_1^{\xi_{t\ arg\ et}})), (s(o_x, t_{t\ arg\ et}, l_2^{\xi_{t\ arg\ et}})), \cdots\rangle$ which correspond to the spatiotemporal point $(t_{opt}, l_i^{\xi_{opt}})$, until $s(o_x, t_{t\ arg\ et}, l_i^{\xi_{t\ arg\ et}}) = 1$, then return the location $l_{t\ arg\ et} = l_i^{\xi_{t\ arg\ et}}$ of the vehicle o_x at the moment $t_{t\ arg\ et}$.

$$t_{opt} = \underset{t_{opt} \in (t_s, t_{t\ arg\ et})}{\arg\min} \left(Cost(t_{opt}, \xi_{opt}) + \sum_{i=1}^{|L|} p_{l_i}^{opt} \times Cost(t_{t\ arg\ et}, \xi_{t\ arg\ et}) \right) \tag{10}$$

4.3.3. Intermediate Searching at Heuristic Moments (IHMs). Calculate the probability $\{(p_{l_1}^{opt}, p_{l_2}^{opt}, \cdots)\}$ of o_x moving to

Table 2: Accumulative search cost of QDP with different learning rates.

Start moment	Training time	8:00	10:00	12:00	14:00	16:00	18:00	20:00
$\alpha=1.0,\gamma=1$	**11.50** h	32.81	32.90	34.85	32.72	34.06	30.15	34.97
$\alpha=0.5,\gamma=1$	**30.36** h	32.81	32.90	34.85	32.72	34.06	30.15	34.97
$\alpha=0.3,\gamma=1$	**54.12** h	32.81	32.90	34.85	32.72	34.06	30.15	34.97
$\alpha=0.1,\gamma=1$	**98.34** h	32.81	32.90	34.85	32.72	34.06	30.15	34.97

Table 3: Accumulative search cost of QDP with different discount rates.

Start moment	8:00	10:00	12:00	14:00	16:00	18:00	20:00
$\alpha=1,\gamma=0.97$	34.38	35.05	36.51	34.86	35.57	32.44	36.61
$\alpha=1,\gamma=0.99$	33.25	33.42	35.01	33.27	34.31	30.46	35.11
$\alpha=1,\gamma=1.00$	**32.81**	**32.90**	**34.85**	**32.72**	**34.06**	**30.15**	**34.97**
$\alpha=1,\gamma=1.01$	33.59	32.95	34.91	33.08	33.72	30.18	35.14
$\alpha=1,\gamma=1.03$	33.15	33.19	35.79	33.24	34.10	30.36	35.13
$\alpha=1,\gamma=1.05$	33.68	33.64	36.17	33.92	34.96	31.21	35.79

different locations at moment $\{t_{opt}\}, t_{opt} \in (t_s, t_{t\ \arg\ et}]$ according to the spatiotemporal point (t_s, l_s) and the historical trajectories TR'; Determine the searching location sequence $\{\langle t_{opt}, \xi_{opt}\rangle\}$ according to the location decision model; Determine the intermediate search moment t_{opt} according to Equation (11); Perform spatiotemporal searches $\langle(s(o_x, t_{opt}, l_1^{\xi_{opt}})), (s(o_x, t_{opt}, l_2^{\xi_{opt}})), \cdots\rangle$, until $s(o_x, t_{opt}, l_i^{\xi_{opt}}) = 1$; Update spatiotemporal point $(t_s, l_s) = (t_{opt}, l_i^{\xi_{opt}})$; Iterate the above steps until $t_s = t_{t\ \arg\ et}$, and return the location $l_{t\ \arg\ et} = l_i^{\xi_{opt}}$ of the vehicle o_x at the target time $t_{t\ \arg\ et}$.

$$t_{opt} = \underset{t_{opt} \in (t_s, t_{t\ \arg\ et}]}{\arg\min} \frac{Cost(t_{opt}, \xi_{opt})}{t_{opt} - t_s} \tag{11}$$

In summary, ALT searches only at the target moment, IEM searches at an intermediate moment and the target moment, while IHMs and QDP search at multiple intermediate moments as well as the target moment. The baselines and QDP use the same location decision model as described in Section 3.1, which is not the focus of this paper. The main difference among them is how to decide the searching moment.

4.4. Comparison of Different Algorithms. This experiment compares four spatiotemporal searching algorithms: ALT, IEM, IHMs and QDP. When testing, instead of real camera records, we use the testing trajectories as the source of ground truth, which also can enable the functionality of Equation (1). The performance metric is the average search cost of the 1000 tests. As shown in Figure 6, QDP is better than the other three spatiotemporal searching algorithms in general. The four spatiotemporal searching algorithms have different performance under different grid sizes (1KM × 1KM, 1.2KM × 1.2KM, and 1.5KM × 1.5KM, respectively) and time intervals (i.e., $t_{t\ \arg\ et} - t_s$= 30 min, 40 min, 50 min, respectively). The start moment ts traverses every moment from 7:00 to "21:59 - time_span" of the corresponding testing day.

(1) When the grid size is 1KM × 1KM, IHMs and QDP are better than ALT and IEM at different time intervals. As the time interval becomes bigger, the advantage of QDP is increasing compared with IHMs

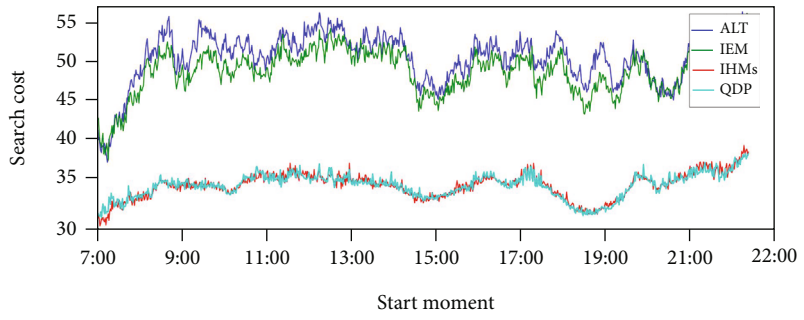
(2) When the grid size is 1.2KM × 1.2KM, IHMs and QDP are better than ALT when the time interval is 30 min or 40 min. There is no significant difference among ALT, IHMs, and QDP when the time interval is 50 min, but they are better than IEM

(3) When the grid size is 1.5KM × 1.5KM, IHMs and QDP are better than ALT when the time interval is 30 min. There is no significant difference among ALT, IHMs, and QDP when the time interval is 40 min or 50 min, but they are better than IEM

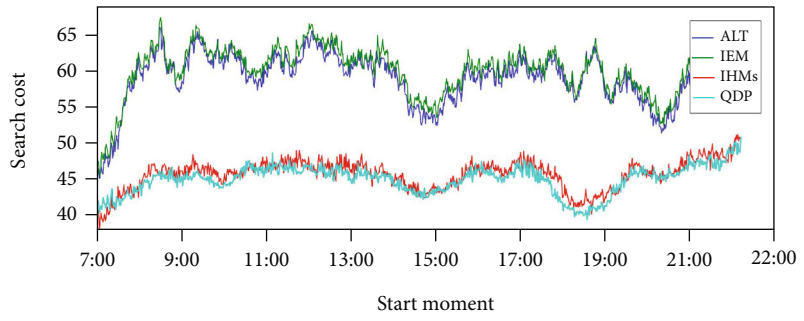
No matter what the grid size and time interval is, QDP is always better than or equal to IHMs.

4.5. Analysis and Discussion. We perform more analysis to disclose the reason of the above results. Remind that the main difference among ALT, IEM, IHMs, and QDP is how to decide the searching moment t_{opt}, i.e., ALT searches only at the target moment, IEM searches at an intermediate moment and the target moment, while IHMs and QDP search at multiple intermediate moments as well as the target moment. We intend to demonstrate the efficiency of different moment decisions. Because the search cost will increase as the timespan $\Delta t = t_{opt} - t_s$ becomes larger, the cost-timespan ratio (also the heuristic indicator of IHMs) is defined as the ratio of the estimated search cost to the timespan:

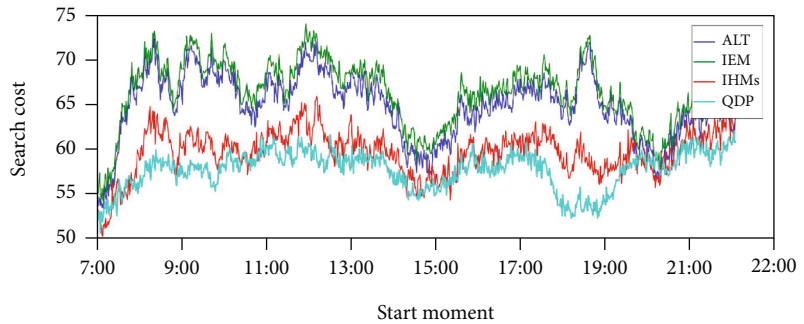
$$CostTimespanRatio(t_{opt} - t_s) = \frac{Cost(t_{opt}, \xi_{opt})}{t_{opt} - t_s} \tag{12}$$



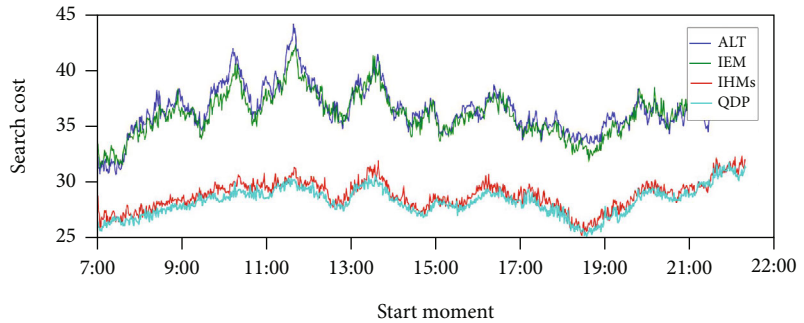
(a) grid size = 1KM × 1KM, target moment - start moment = 30 min



(b) grid size = 1KM × 1KM, target moment - start moment = 40 min

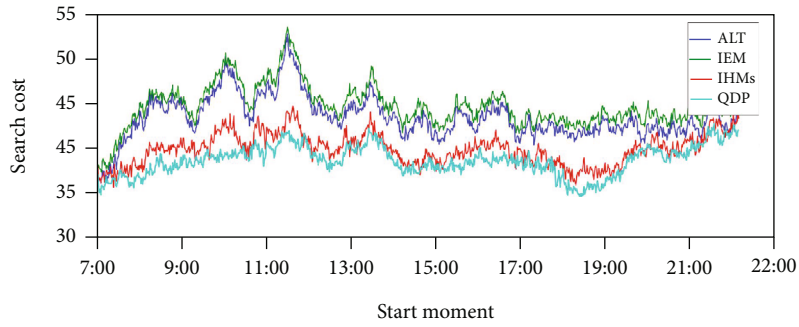


(c) grid size = 1KM × 1KM, target moment - start moment = 50 min

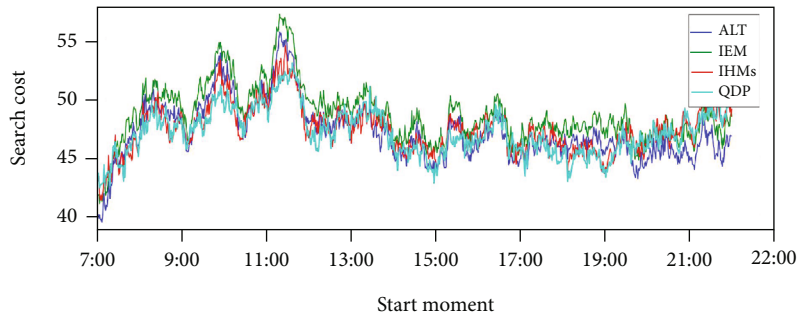


(d) grid size = 1.2KM × 1.2KM, target moment - start moment = 30 min

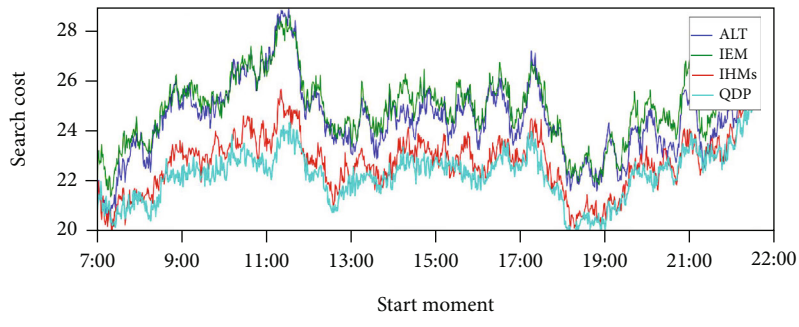
Figure 6: Continued.



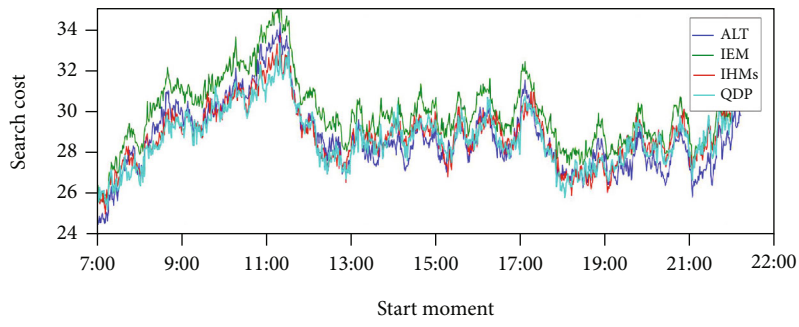
(e) grid size = 1.2KM × 1.2KM, target moment - start moment = 40 min



(f) grid size = 1.2KM × 1.2KM, target moment - start moment = 50 min

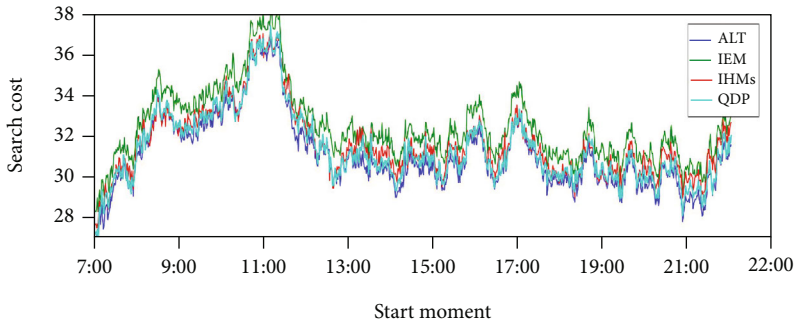


(g) grid size = 1.5KM × 1.5KM, target moment - start moment = 30 min



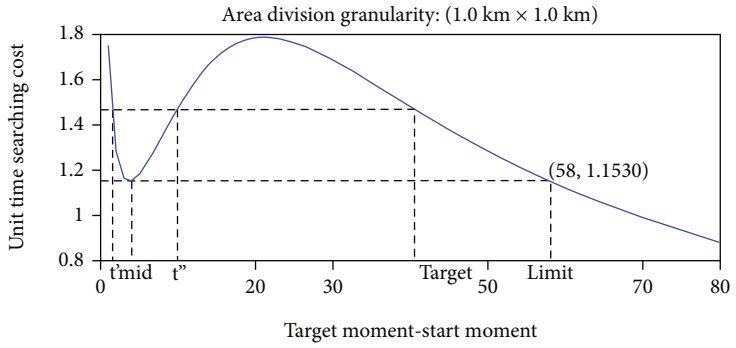
(h) grid size = 1.5KM × 1.5KM, target moment - start moment = 40 min

Figure 6: Continued.

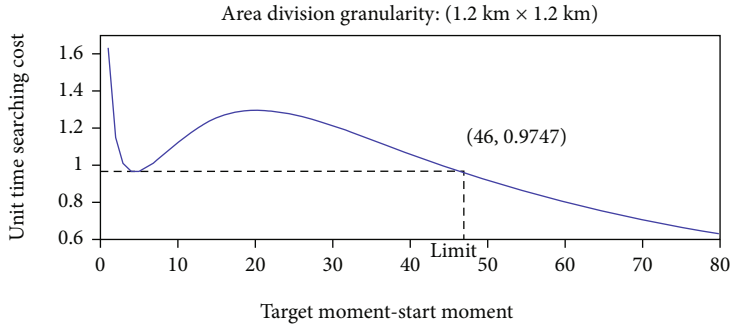


(i) grid size = 1.5KM × 1.5KM, target moment - start moment = 50 min

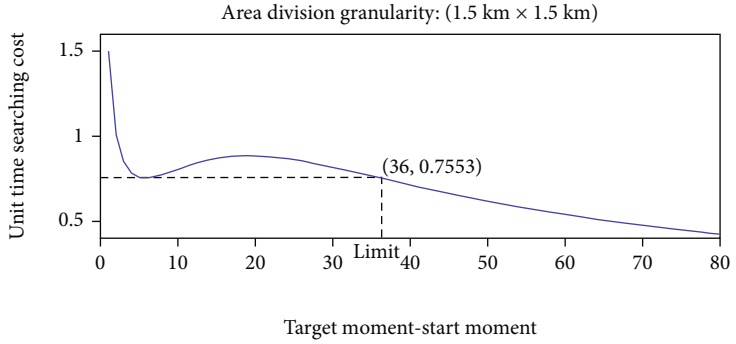
Figure 6: Comparison of four spatiotemporal searching algorithms.



(a) grid size = 1KM × 1KM



(b) grid size = 1.2KM × 1.2KM



(c) grid size = 1.5KM × 1.5KM

Figure 7: Cost-timespan ratio of different timespans.

Figure 7 shows the cost-timespan ratio under different grid sizes (1KM × 1KM, 1.2KM × 1.2KM, 1.5KM × 1.5KM). All three ratio's curves first decrease, then increase, and then decrease along with the timespan. Let preferred timespan Δt_{mid} be the timespan of the first locally minimum ratio CostTimespanRatio(Δt_{mid}). A contour timespan $\Delta t_{\lim it}$ is the timespan whose cost-timespan ratio is equal to CostTimespanRatio(Δt_{mid}).

If $\Delta t < \Delta t_{mid}$ or $\Delta t > \Delta t_{\lim it}$, the larger timespan is more efficient, thus the searching moment should be as later as possible, i.e., searching at intermediate moments cannot help to reduce the total search cost, and ALT will be better than IEM. In this situation, IHMs and QDP will also only perform searches at the target moment, which are equivalent with ALT.

If Δt is between (Δt_{mid}, $\Delta t_{\lim it}$), since CostTimespanRatio(Δt_{mid}) is the minimum ratio when timespan is from 0 to Δt. As a result, Δt_{mid} is the most efficient timespan which can decide the corresponding intermediate moment, i.e., searching at intermediate moments can reduce the total search cost, and IEM, IHMs and QDP may be better than ALT. Whether the right intermediate moments are detected is up to specific algorithms.

The different grid sizes will cause $\Delta t_{\lim it}$ to be different.$\Delta t_{\lim it}$ corresponding to grid sizes 1KM × 1KM, 1.2KM × 1.2KM, and 1.5KM × 1.5KM are calculated to be 58 min, 46 min and 36 min, respectively. Therefore, the reason of the results in Section 4.4 can be explained as:

(1) When the grid size = 1KM × 1KM, In Figures 6(a), 6(b), 6(c), since the timespans Δt = 30 min/40 min/50 min do not exceed $\Delta t_{\lim it}$ (58 min), IHMs and QDP are better than ALT

(2) When the grid size = 1.2KM × 1.2KM, Since the timespans Δt = 30 min/40 min (Figures 6(d), 6(e)) do not exceed $\Delta t_{\lim it}$ (46 min), IHMs and QDP are better than ALT. While when the timespan Δt = 50 min (Figure 6(f)) exceeds $\Delta t_{\lim it}$ (46 min), there is no significant difference among ALT, IHMs, and QDP

(3) When the grid size = 1.5KM × 1.5KM, Since the timespan Δt = 30 min (Figure 6(g)) does not exceed $\Delta t_{\lim it}$ (36 min), IHMs and QDP are better than ALT. While when the timespans Δt = 40 min/50 min exceed $\Delta t_{\lim it}$ (36 min), there is no significant difference among ALT, IHMs, and QDP

IEM is bound to search at an intermediate moment and cannot make a decision dynamically to adapt environment information. Therefore, IEM is almost the worst in all situations. QDP can surmount the shortage that IHMs may output a locally optimal solution. As Q-learning does, it finds the near global optimal solution in the sense that it minimizes the total cost of any and all successive searches.

5. Conclusions

To minimize the cost of spatiotemporal search, this paper proposes a reinforcement learning algorithm called QDP.

QDP selects the next searching moment based on known vehicle's spatiotemporal point. The outcome of these searches is a new known vehicle's spatiotemporal point, which guides the next selection iteratively. To address the challenge of probabilistic state in the training phase, we propose a novel training method for QDP, which is based on Monte-Carlo method and probabilistic states. QDP replaces the next state by multiple states of a probability distribution and estimates the expected cost of subsequent actions to calculate the value function. Finally QDP creates a state and an action randomly in each loop to train the value function progressively.

We evaluate QDP on a real-world vehicle trajectory data and compare it with baseline algorithms ALT, IEM, IHMs under different grid sizes (1KM × 1KM, 1.2KM × 1.2KM, 1.5KM × 1.5KM) and different time intervals (30 min, 40 min, 50 min). The experimental results show that QDP is better than the baseline. In addition, with the grid size decreasing, the preferred timespan Δt_{mid} is also decreasing, but the contour timespan $\Delta t_{\lim it}$ is increasing. Compared with ALT, algorithms with adaptive intermediate moment selection (i.e., IHMs and QDP) can effectively reduce the total search cost when the timespan is between Δt_{mid} and $\Delta t_{\lim it}$. Compared with IHMs, QDP takes into account both the cost efficiency of the current selection (as well as performing search at that moment) and its potential impact on subsequent selections, and gives an approximate lower bound of the cost of spatiotemporal search to a certain extent.

In the future, we will test QDP with other datasets, e.g., trajectories of pedestrians, bicycles, or ride sharing cars [34, 35]. We also wish to build new models that can take consideration of both the location decision model and the moment decision model simultaneously, or can output directly a candidate spatiotemporal point to search for next. And deep reinforcement learning algorithms are worth to explore for this problem.

Notations

d_j:	a day, $d_j \in D$; d_x is the testing day
o_x:	the target vehicle, $o_x \in O$
l_i:	a location
t_k:	a moment, $d_j = \langle t_1, t_2, \cdots, t_{\lvert d_j \rvert} \rangle$
(t_s, l_s):	a known spatiotemporal point
t_{opt}:	the intermediate search moment
$t_{t \arg et}$:	the target moment
$l_{t \arg et}$:	o_x's location at $t_{t \arg et}$
$tr(o_x, d_j)$:	the trajectory of o_x in d_j, $tr(o_x, d_j) \in$TR
TR':	the training trajectories
$c_{l_i}^{opt}$:	the cost of searching at (t_{opt}, l_i)
$\langle t_{opt}, \xi_{opt} \rangle$:	the searching location sequence at moment t_{opt}, i.e., the location decision
$l_i^{\xi_{opt}}$:	o_x's location at t_{opt}, assume it is the *i-th* location of $\langle t_{opt}, \xi_{opt} \rangle$
$p_{l_i}^{opt}$:	the probability that o_x will be at l_i at t_{opt}

$TPM_{\Delta t}^{|L|\times|L|}$: the transfer probability matrix; an element is $p_{l_i}^{opt}$

$Cost(t_{opt}, \xi_{opt})$: the expected search cost of $\langle t_{opt}, \xi_{opt} \rangle$

$S_{l_s}^s$: a state, also denoted as (t_s, l_s)

$A_{l_s}^s$: an action performed under $S_{l_s}^s$, i.e., $\langle t_{opt}, \xi_{opt} \rangle$

$S_{l_i^\xi}^{opt}$: a probabilistic state after $A_{l_s}^s$, also denoted as $(t_{opt}, l_i^{\xi_{opt}})$

$A_{l_i^\xi}^{opt}$: an action performed under $S_{l_i^\xi}^{opt}$.

Data Availability

The experimental dataset[The experimental dataset of this article is provided on demand, please contact the author if necessary.] is derived from the trajectories of 19,000 taxis in Chengdu, China, in August 2014. The data acquisition area is about, and the duration of each trajectory is from 7 : 00 am to 21 : 59 pm.

Conflicts of Interest

Please note the author declared that there is no conflict of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under grant No. 61772136, and the Fujian Engineering Research Center of Big Data Analysis and Processing.

References

[1] X. Liu, W. Liu, T. Mei, and H. Ma, "A deep learning-based approach to progressive vehicle Re-identification for urban surveillance," in *ECCV 2016: Computer Vision – ECCV 2016*, pp. 869–884, Springer, 2016.

[2] Y. Shen, T. Xiao, H. Li, S. Yi, and X. Wang, "Learning deep neural networks for vehicle re-id with visual-spatio-temporal path proposals," in *IEEE International Conference on Computer Vision*, Venice, Italy, 2017.

[3] Y. Jin, C. Li, Y. Li, P. Peng, and G. A. Giannopoulos, "Model latent views with multi-center metric learning for vehicle re-identification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1919–1931, 2021.

[4] X. Liu, W. Liu, T. Mei, and H. Ma, "PROVID: Progressive and Multimodal Vehicle Reidentification for Large-Scale Urban Surveillance," *IEEE Transactions on Multimedia*, vol. 20, no. 3, pp. 645–658, 2018.

[5] Z. Yu, L. Han, C. Chen, W. Guo, and Z. Yu, "Object Tracking by Least Spatiotemporal Searches," vol. 8, no. 16, pp. 12934–12946, 2020.

[6] Puterman and L. Martin, *Markov Decision Process. Simulation-Based Algorithms for Markov Decision Processes*, Springer, London, 2013.

[7] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[8] F. S. Melo and M. I. Ribeiro, "Convergence of Q-learning with linear function approximation," in *2007 European Control Conference (ECC)*, Kos, Greece, 2007.

[9] B. O. Koopman, "The theory of search. I. Kinematic bases," *Operations Research*, vol. 4, no. 3, pp. 324–346, 1956.

[10] B. O. Koopman, "The theory of search. II. Target detection," *Operations Research*, vol. 4, no. 5, pp. 503–531, 1956.

[11] B. O. Koopman, "The Theory of search," *Operations Research*, vol. 5, no. 5, pp. 613–626, 1957.

[12] L. D. Stone, *Theory of Optimal Search*, Academic Press, 1975.

[13] L. D. Stone, "Necessary and sufficient conditions for optimal search plans for moving targets," *Mathematics of Operations Research*, vol. 4, no. 4, pp. 431–440, 1979.

[14] J. Berger, N. Lo, and M. Noel, "Exact Solution for Search-and-Rescue Path Planning," *International Journal of Computer and Communication Engineering*, vol. 2, pp. 266–271, 2013.

[15] N. Lo, J. Berger, and M. Noel, "Toward optimizing static target search path planning," in *2012 IEEE Symposium on Computational Intelligence for Security and Defence Applications*, Ottawa, ON, Canada, 2012.

[16] K. S. Tseng and B. Mettler, "Near-optimal probabilistic search via submodularity and sparse regression," *Autonomous Robots*, vol. 41, no. 1, pp. 205–229, 2017.

[17] J. Berger and N. Lo, "An innovative multi-agent search-and-rescue path planning approach," *Computers & Operations Research*, vol. 53, pp. 24–31, 2015.

[18] E. Bensana, G. Verfaillie, J. C. Agnese, N. Bataille, and D. Blumstein, "Exact &INEXACT Methods for Daily Management of Earth Observation Satellite," *Space Mission Operations and Ground Data Systems-SpaceOps' 96*, vol. 394, p. 507, 1996.

[19] E. Bensana, M. Lemaitre, and G. Verfaillie, "Earth observation satellite management," *Constraints*, vol. 4, no. 3, pp. 293–299, 1999.

[20] T. Benoist, "Towards optimal formwork pairing on construction sites," *RAIRO - Operations Research*, vol. 41, no. 4, pp. 381–398, 2007.

[21] M. Lemaître, G. Verfaillie, F. Jouhaud, J. M. Lachiver, and N. Bataille, "Selecting and scheduling observations of agile satellites," *Aerospace Science and Technology*, vol. 6, no. 5, pp. 367–381, 2002.

[22] F. Wörgötter and B. Porr, "Reinforcement learning," *Approximate Dynamic Programming & Reinforcement Learning*, vol. 3, no. 3, pp. 804–809, 2008.

[23] A. Plaat, *Deep Reinforcement Learning*, Springer Nature Singapore, 2022.

[24] A. Jevtić, A. Colomé, G. Alenyà, and C. Torras, "Robot motion adaptation through user intervention and reinforcement learning," *Pattern Recognition Letters*, vol. 105, pp. 67–75, 2018.

[25] A. W. Moore, "Variable Resolution Dynamic Programming: Efficiently Learning Action Maps in Multivariate Real-valued State-spaces," in *Machine Learning Proceedings 1991*, pp. 333–337, Evanston, Illinois, 1991.

[26] R. H. Crites and A. G. Barto, "Elevator group control using multiple reinforcement learning agents," *Machine Learning*, vol. 33, no. 2/3, pp. 235–262, 1998.

[27] D. Silver, A. Huang, C. J. Maddison et al., "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[28] D. Silver, J. Schrittwieser, K. Simonyan et al., "Mastering the game of Go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[29] B. J. A. Kroese, "Learning from delayed rewards," *Robotics and Autonomous Systems*, vol. 15, no. 4, pp. 233–235, 1995.

[30] G. A. Rummery and M. Niranjan, *On-Line Q-Learning Using Connectionist Systems*, University of Cambridge, Cambridge, Department of Engineering, 1994.

[31] H. van Seijen, H. van Hasselt, S. Whiteson, and M. Wiering, "A theoretical and empirical analysis of Expected Sarsa," in *2009 IEEE symposium on adaptive dynamic programming and reinforcement learning*, pp. 177–184, Nashville, TN, USA, 2009.

[32] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Playing atari with deep reinforcement learning," 2013, http://arxiv.org/abs/1312.5602.

[33] J. Chen, *One-way optimal search theory (in Chinese)*, National Defense Industry Press, 2016.

[34] C. Chen, Y. Ding, X. Xie, S. Zhang, Z. Wang, and L. Feng, "TrajCompressor: an online map-matching-based trajectory compression framework leveraging vehicle heading direction and change," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 5, pp. 2012–2028, 2020.

[35] S. Guo, C. Chen, J. Wang et al., "ROD-revenue: seeking strategies analysis and revenue prediction in ride-on-demand service using multi-source urban data," *IEEE Transactions on Mobile Computing*, vol. 19, no. 9, pp. 2202–2220, 2019.