Route Planning Based on Parallel Optimization in the Air-Ground Integrated Network

Ken Cai, Tianlun Dai, Qinyong Lin, Xinyang Song, Wei Zhou, Qian Zhou, Jinzhan Wei, Feng Wang, Huazhou Chen, and Bohan Li

***Abstract*— Recent advancement in propulsion technologies to reduce the need for travel or increase the share of sustainable unmanned devices has accelerated the shift toward sustainable transport. To achieve the optimization of route planning in the air-ground integrated network (AGIN), we design an optimization strategy of accompanying graph navigation for unmanned devices, which aims to reduce the power consumption and CO_2 gas emissions. The optimization of accompanying graph navigation is composed of three strategies, namely, the navigation based on the complete maps, the navigation based on the partitioned maps, and the navigation without maps. We propose a Two-tiered Grid (TG) index and Distributed AGIN Navigation (DAN) to navigate on partitioned maps. The top layer of the TG-index is composed of the border vertices of the global road network, which reflects the overall traffic conditions of the global road network and provides coarse-grained navigation routes. The bottom layer is a grid index composed of subgraphs, which reflects traffic conditions in local areas and provides fine-grained navigation routes. The navigation optimization is implemented in several segments, which can be run by multi-processors and realize rapid response to a large number of concurrent queries.**

***Index Terms*— Route planning, parallel, air-ground integrated network, sustainable transport.**

Manuscript received 4 July 2022; revised 27 September 2022, 17 November 2022, and 24 December 2022; accepted 29 December 2022. Date of publication 16 January 2023; date of current version 29 November 2023. This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFB1708100, in part by the National Natural Science Foundation of China under Grant 62003379, in part by the "14th Five-Year Plan" Civil Aerospace Pre-research Project of China under Grant D020101, in part by the Guangzhou Science and Technology Program under Grant 202201011274, in part by the Special Projects in Key Areas of Guangdong's Colleges under Grant 2021ZDZX4061 and Grant 2022ZDZX3007, in part by the Research Project and Development Plan for Key Areas of Guangdong Province under Grant 2020B0202080002, and in part by the CCF-Huawei Database System Innovation Research Plan under Grant CCF-HUAWEIDBIR2020001A. The Associate Editor for this article was Z. Lv. *(Ken Cai and Tianlun Dai contributed equally to this work.) (Corresponding author: Qinyong Lin.)*

Ken Cai and Qinyong Lin are with the College of Automation, Zhongkai University of Agriculture and Engineering, Guangzhou 510225, China (e-mail: icken@zhku.edu.cn; linqinyong@foxmail.com).

Tianlun Dai, Xinyang Song, Wei Zhou, and Feng Wang are with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China (e-mail: sx2016060@nuaa.edu.cn; sx2016017@nuaa.edu.cn; 157472129@qq.com; 434372488@qq.com).

Qian Zhou is with the School of Modern Posts, Nanjing University of Posts and Telecommunications, Nanjing 210003, China (e-mail: zhouqian@njupt.edu.cn).

Jinzhan Wei is with the School of Electronic Information and Automation, Guilin University of Aerospace Technology, Guilin 541010, China (e-mail: 17468297@qq.com).

Huazhou Chen is with the College of Science, Guilin University of Technology, Guilin 541000, China (e-mail: chenhz@glut.edu.cn).

Bohan Li is with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China, and also with the Key Laboratory of Safety-Critical Software, Ministry of Industry and Information Technology, National Engineering Laboratory for Integrated Aero-Space-Groud-Ocean Big Data Application Technology, Qingdao 266061, China (e-mail: bhli@nuaa.edu.cn).

Digital Object Identifier 10.1109/TITS.2023.3234936

I. Introduction

ROUTE planning of unmanned devices like UAVs (Unmanned Aerial Vehicles) is crucial for location-based services (LBSs). The increasing number of vehicles in sustainable transport systems rely on navigation routes to avoid congestion and reach their destination as soon as possible. Although mainstream navigation systems can update the navigation route for the driving users according to the road conditions, considering the calculation cost and driving habits, these kinds of navigation systems cannot update the navigation route for the users in time according to the frequently varying road conditions. In fact, the time consumption of travel is usually changing. Therefore, the road network is called a dynamic road network. It is not easy to obtain the best planning path in real-time on the dynamic road network, because it produces a huge computational cost when dealing with a great number of concurrent navigation queries on the large-scale road network (the number of vertices usually exceeds 1 million). Then, how to optimize and respond to each navigation query in real time is a challenging problem.

The problem of route planning has received continuous attention [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], but the above challenges have not been completely solved. In the past, many works have been studying the route problem in the field of changing road conditions. The shortest route is calculated according to the driving time. In order to ensure that the provided path is the shortest in all candidate paths, the shortest path must be repeatedly calculated over the changing conditions of the road, which causes huge calculation costs. In addition, most of the existing works focuses on dealing with a single route planning query, and less on dealing with plenty of queries that happen at the same time. Compared with the accurate calculation of repeatedly calculating the shortest route from the source to the target position, it is more feasible to adaptively adjust the similar best path which can meet the changing road conditions in a very short time. Based on this idea and considering plenty of concurrent queries generate at the same time, a parallel path query algorithm that uses a two-tiered grid index is proposed in order to realize

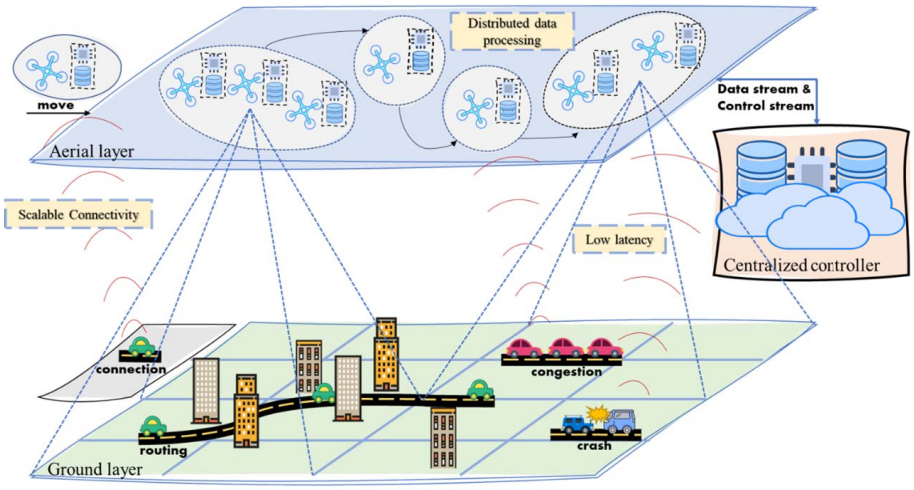


Fig. 1. The architecture of the air-ground integrated network.

the parallel optimization of navigation with the Master-Worker paradigm.

Our air-ground integrated network architecture, as shown in Fig 1, is a deep integration of three-layer structures: cloud computing layer, edge computing layer, and device layer to achieve low latency, high reliability, and low power consumption.

- Cloud computing layer: It provides basic computing functions, and adopts a layered computing architecture to handle computing task allocation and resource management of different scales.
- Edge computing layer: It is more suitable for distributed local processing of small-scale delay-sensitive computing tasks, which can be deployed close to users and distributed at the network edge. The UAVs support the local route planning part of the DAN algorithm.
- Device layer: It consists of aerial and ground devices, including UAVs, vehicles, base stations, routers, and other IoT devices. Ground devices support the Trust Model, and aerial devices UAVs can communicate with ground unmanned vehicles and provide distributed data processing capabilities.

There are three main contributions to this research:

1) Based on the air-ground integrated network, a parallel route planning algorithm is proposed to transform some continuous navigation route optimization problems of the entire road network into the local optimal route problems of multiple sub-networks, which reduces the computational complexity, improves the response speed and continuously optimizes the navigation route in parallel.
2) A two-tiered grid index (TG-index) is proposed to provide two-level cascade pruning for the path planning process, which can help to divide some hard missions into easy submission and is easy to realize distributed implementation. The response speed and throughput can be improved by increasing hardware resources so that the whole system has good scalability and reduces the time cost in path planning queries.
3) We conducted extensive experiments to evaluate our proposed model, and the results showed the effectiveness and superiority of our proposal.

The rest of the paper is organized as follows. Section II discusses the related work about the route planning algorithm. The system architecture is present in Section III, and we narrate the proposed architecture from five features of high security, adaptive region of UAV, distributed data processing, low latency, and sustainable and scalable connection. Section IV presents the distributed TG index, and the DAN model is discussed in Section V. Section VI presents the results of experiments, and Section VII concludes this paper.

II. Related Work

The route planning algorithm is the key technology to realize the autonomous navigation of unmanned vehicles. Considering the complex reality of road situations, adjusting the navigation path without time-lapse can avoid heavy traffic in advance and unnecessary time consumed to a great extent which is overly critical in the current navigation path service [15], [16]. An excellent route planning algorithm can provide an efficient path in a complex environment, which can improve the working efficiency and reduce the loss of the unmanned vehicle [1]. Route planning algorithms are generally divided into three types, namely traditional planning algorithms, intelligent planning algorithms, and multi-algorithm fusion planning algorithms.

Traditional planning algorithms need to be modeled and calculated in a structured environment. Khatib proposed the Artificial Potential Field (APF) method [17]. Its basic idea is to design the motion of the robot in the surrounding environment as an abstract artificial gravitational field. The target point has a "gravitational force" on the mobile robot, and the obstacle has a "repulsive force" on the mobile robot. Finally, by seeking the resultant force to control the movement of the mobile robot. The path planned by the APF is generally smooth and safe, but this method has the problem of local optimum. Li et al. [18] eliminated the local optima problem by setting a virtual local objective. Liu et al. [19] proposed a two-potential-fields fused adaptive path planning system, which fuses the

horizontal velocity field constructed by velocity information with the artificial potential field so that the system can adapt to different speeds and different types of obstacles. Another traditional route planning algorithm is the Grid Map (GM) method. The Dijkstra algorithm [20] was proposed in 1959. It can calculate the shortest distance between two vertices on an undirected graph. However, as the number of nodes increases, the computational efficiency decreases. Zhang et al. [21] improved the Dijkstra algorithm, using the start point and end point of the line as the two vertices of the diagonal of the rectangular area, and only traversing the grid within the rectangular range, reducing meaningless time-consuming searches. Chen et al. [22] changed the storage structure, and the binary sorting tree algorithm based on a rectangular restricted area reduced the memory storage space and shortened the shortest path calculation time. Hart et al. [23] proposed the A* algorithm in 1972. This algorithm combines the advantages of Dijkstra's algorithm. While performing heuristic search to improve the efficiency of the algorithm, it can guarantee to find an optimal path. However, this method requires a lot of computation, and the planned path is often more inflected. Wang et al. [9] improved the A* algorithm and proposed the R3 system, which uses historical data to select a path in a certain direction, which can save large computing time. Different from the R3 system, Demiryurek et al. [8] divided the road network into non-overlapping partitions and proposed a bidirectional search strategy to implement an efficient A* algorithm. Min et al. [24] considered the cost of path curvature in the heuristic function to improve the smoothness of the path. Although many scholars have made different improvements to traditional planning algorithms, most of the methods are designed based on a single-computer computing environment. When faced with large-scale online queries, the computing power is insufficient, and the scalability is poor. Sharma et al. [25] proposed a secure information management scheme, Third Eye, which satisfies all the performance parameters as well as satisfies the trust metrics among the vehicles and devices.

Intelligent planning algorithms have a certain learning ability and optimize the path by acquiring new information while planning [26], [27]. Colorni et al. [28] proposed the Ant Colony Optimization (ACO). Although the ACO has high robustness, it is prone to local optimal problems. Sangeetha et al. [29] proposed a gain-based intelligent ant colony optimization algorithm, which reduces energy by eliminating all paths that need to traverse around obstacles, thereby improving the efficiency of the ACO. The work [30], [31] mainly made up for the defects of ACO through algorithm fusion. Bremermann et al. [32] proposed a genetic algorithm, which can search different regions of the solution space in parallel, but the algorithm has a low operation rate and takes up a lot of computing resources during the operation. Qu et al. [33] used a co-evolutionary mechanism to improve the genetic algorithm, thereby avoiding the local optimum problem and making the convergence faster. Nazarahari et al. [34] designed a new artificial potential field algorithm, which finds all feasible paths between the start point and the endpoint in a grid environment and uses five crossover and mutation operators to solve the problem that the initial solution of the traditional genetic algorithm cannot obtain the optimal solution after iterating for many times. Although many scholars have made some improvements in intelligent planning algorithms, there are still problems such as easy falling into local optimum, low operation rate, and large memory usage. Based on PM2.5 retrieval with high spatiotemporal resolution and a dynamic Dijkstra algorithm, Gao et al. [35] proposed a short-distance healthy route planning approach. Woźniak et al. [36] proposed a type-2 fuzzy system to share knowledge of road conditions and evaluate better all the features of driving with tolerance to the style of driving of different people. Kumar et al. [37] proposed a method to identify reflected signals using range data, which efficiently improves GPS position accuracy.

The multi-algorithm fusion planning algorithm usually combines some existing technologies to speed up the route planning process. Designing an effective index structure is one of the technologies to accelerate route planning [38], [39]. Wu et al. [38] proposed a Neural Index Search framework, whose core is to train a search policy to enhance the performance of the query. Some recent works provided continuous route planning services for navigation users considering varying road conditions. Dai et al. [40] proposed a continuous route planning algorithm based on the pruning strategy, which calculates the local optimal path through the greedy principle, which significantly reduces the cost of computing for the shortest path on the entire graph, and the planned paths can be adjusted in real-time according to changing traffic conditions. The work [41] proposed the PARP algorithm. The PARP algorithm considers the maintenance cost of the dynamic road network and presents DLP index structure. At the same time, the complete road network is divided into multiple subnets, and the route planning query process is accelerated on the basis of the distributed architecture. Researchers also try to solve the planning path by mixing the present and coming information together [42], [43], [44], [45], these methods can work better with plenty of data, which needs lots of drivers obeying the dispatching. Although the multi-algorithm fusion planning method makes up for the shortcomings of the existing work to a certain extent, it uses multiple computing nodes to calculate the same navigation query using a serial calculation method, which is difficult to greatly improve search efficiency.

III. System Architecture

Some system overview and details will be shown in this part. Fig. 2 shows the framework of the system, our system provides LBS based on air-ground integrated network. Our system adopts the trust model in the paper [46], and introduces the TG-index and parallel route planning algorithm.

A. High Security

The trust model proposed in the paper [46] is used to identify and remove LBS requests sent by malicious vehicles, which improves the security of the system.

B. Adaptive Region of UAV

UAV resources can be dynamically allocated to avoid the problems of load balancing and insufficient connectivity

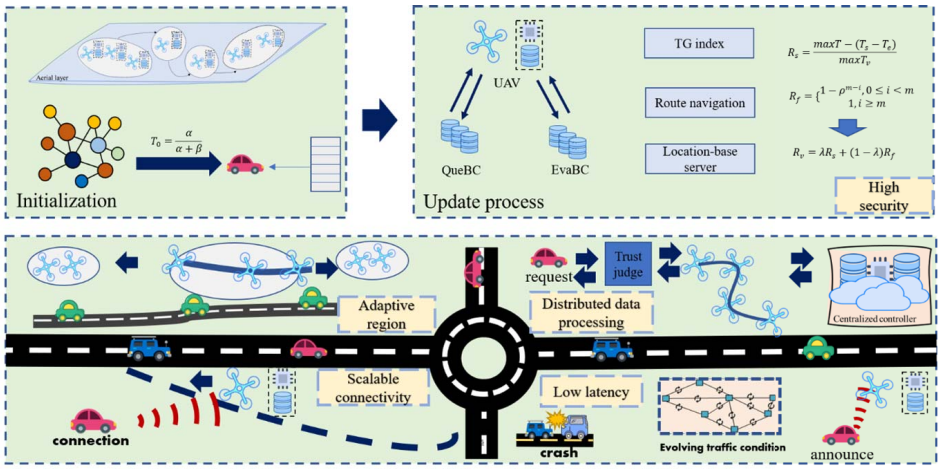


Fig. 2. The procedure and features of the air-ground integrated network.

caused by excessive vehicle service requests, virtualize UAV resources, and perform adaptive allocation. The layered framework can flexibly adjust the granularity of resource allocation, especially the load allocation.

C. Distributed Data Processing

The computing demand is not invariable. It can adjust itself according to the real conditions in the air-ground system, and the connections between UAV can speed up the process of data handling capacity. With the capability of distributed data processing, we construct the distributed TG-index and can handle the route planning in parallel.

D. Low Latency

UAVs and vehicles can use their mobile characteristics to provide computing service capabilities on demand. In this sense, scalable large-scale connections can be supported through scheduling and dynamic connection management. The air-ground integrated network facilitates intelligent transportation system, like driverless car, which not just receives messages from outside sensors but can still use their own sensors to interact with outside world. At the same time, the warnings can be broadcast to vehicles nearby. Additionally, surveillance in the security field can also be supported by UAV floating connectivity. For instance, when a natural disaster happens, a group of UAV can be sent to the scene. By using the real-time images and video from these UAV, specialized persons can make active decisions.

E. Sustainable and Scalable Connection

UAVs are divided into groups according to vehicle density and establish communication with ground vehicles. UAV areas are adaptively adjusted, and UAV resources can be dynamically allocated to avoid the problem of load balancing and insufficient connectivity caused by excessive vehicle service requests. The UAVs can increase connection capacity on-demand, and as new vehicles enter the air-ground integrated network, new UAVs are dynamically allocated to expand the range of services.

IV. Distributed TG-Index

A. TG-Index Frame

When designing TG-Index, the following three requirements must be satisfied.

1) **Reasonable maintenance cost of the dynamic road network:** indexing shortest paths between selected vertex pairs is very helpful to deal with the process in route planning. However, the weighted edge value is varied all the time and the maintenance cost is high. Therefore, the index structure of the graph must be easy to maintain and minimize overhead as the graph changes dynamically.
2) **Suitable for distributed architectures:** with the growing number of graphs and large-scale concurrent queries, distributed indexes are more scalable and therefore more suitable than centralized indexes. Graph G is partitioned and stored in the cluster, as thus, the index structure of the graph must be maintained independently on different processors.
3) **Supporting parallel algorithms:** for the index, it must quickly assist in the identification of the child map with the relevant query and the distribution of processor chip. Meantime, the reference should have the ability to expand to a great scale, which can satisfy the need of planning path and pruning capacity.

According to the above demands, TG-index is proposed forward. Based on the division of G, TG-index consists of a two-level framework. The first framework is constructed by the k shortest paths among each border vertices. The second framework includes the border vertices of every sub-graph, and an edge connects each pair of border vertices. The minimum weight of k shortest paths is introduced as the weighted value of this edge. TG-index is arranged in the Master-Worker model to realize the ability of parallel route planning.

B. Graph Partition and Determine k Shortest Paths

The construction of the TG-index begins from the partition of graph. Firstly, we partition the graph into several sub-graphs with Metis, and each sub-graph has a number of vertices less than z. Note that each UAV contains a sub-graph.

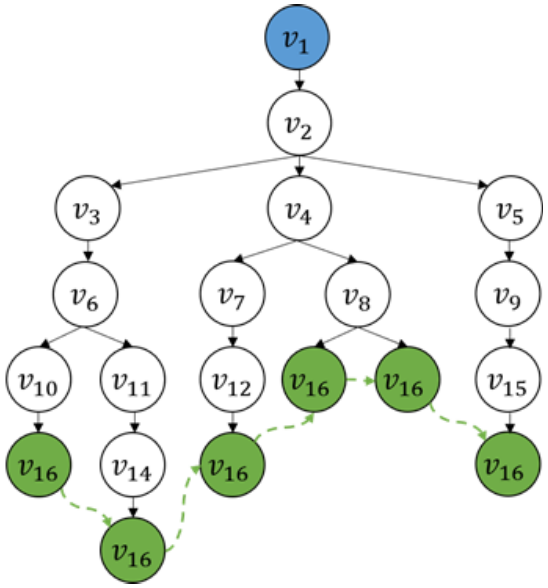


Fig. 3. Prefix path tree.

Definition 1 (Border Vertices): for border vertex $v_x \in V$ exist $v_y \in V$ both satisfy 1) v_x and v_y on the same edge; 2) v_x and v_y are located in two different sub-graphs. Each border vertex enables the realization of path planning in the global schema and supports coarse grain of the path direction. The k-shortest paths of each pair of border vertices are calculated.

The Yen algorithm is an algorithm proposed in 1971 [47]. The Yen algorithm adopts the idea of the deviation path algorithm in the recursive method to solve the k shortest path between two vertices. The same heuristic function is used as the A* algorithm, but the state meaning of this algorithm and the way it expands the state are different.

We obtain k shortest paths with Yen algorithm and store them with Prefix Path Tree (PPT), which is proposed in work [41]. Fig. 3 shows a prefix path tree storing k shortest paths (k = 6) between v_1 and v_{16}. The storage of k shortest paths is only taken as an example here. When we select the value of k, the time cost of creating and maintaining PPT will increase if k is too large, on the contrary, the selection of weight between border vertices will reduce, and the corresponding weight is relatively large, that is, the weight of edge in the skeleton graph is relatively large.

V. DAN Algorithm

A. Framework of DAN Algorithm

DAN (Distributed AGIN Navigation) must have the ability to find out the best path planning of changing graphs persistently and quickly. The forecast about the coming traffic situation is not accurate, the absolute optimal path cannot be calculated for users on the dynamic graph. Therefore, as traffic conditions change, it is not necessary to find out the shortest route from the source to the target position many times. Because the precise shortest route planning will cost much longer time, the DAN takes the similar optimal route as a replacement to speed up the process. This method can satisfy the demands of users and reduce computing power. During route planning, the DAN algorithm first calculates the global path on the skeleton graph, and then continuously optimizes the planned route among the local regions controlled by UAVs.

B. Details of DAN Algorithm

For route planning queries, DAN algorithm adaptively optimizes route planning based on TG-index. The global path p_g from the source vertex to the terminal vertex is calculated based on the G_s, along with the candidate UAVs passing through p_g. Local path optimization is then performed based on the real-time information provided by every UAV while users shift according to the global route. The schematic diagram of DAN is demonstrated in Fig. 4.

Algorithm 1 Computation of Global Path p_g

Require: G_s, $q(v_s, v_t)$, partitioned graphs;
Ensure: the global path p_g from v_s to v_t;
1: Identify SG_s and SG_t by qw_i;
2: **if** v_s and v_t are not border vertices **then**
3: qw_i sends $q(v_s, v_t)$ to gw_s and gw_t;
4: gw_s and gw_t compute ksp from v_s and v_t to each border vertex by Yen;
5: gw_s and gw_t return path information to qw_i;
6: qw_i adds v_s and v_t into G_s;
7: **end if**
8: Search the shortest path from v_s to v_t on G_s as p_g;
9: return p_g;

In order to support parallel processing of route planning queries, TG-index is parallelized in the cluster of Master-Worker mode and the model includes three roles: Master, GridWorker and QueryWorker. The Master is responsible for entire graph G and graph partition. Each QueryWorker maintains a copy of the skeleton graph G_s, while the graph partition is distributed on the GridWorker. The Master will regularly send the varying traffic conditions to all GridWorkers. After receiving the updated weights of edges, GridWorkers will identify the grid firstly, and then immediately update the travel costs between border vertices. After that, GridWorkers send the new travel cost to each QueryWorker, and then update the edge weights on the skeleton graph.

Let v_s and v_t be the source and the destination of the query q. When the query q arrives at a QueryWorker qw_i, it computes the global path p_g and the corresponding pseudocode is shown in Algorithm 1.

C. Local Path Optimization

For a query q (v_s, v_t), where v_s and v_t are source vertex and terminal vertex respectively, the DAN algorithm finds out the approximate shortest optimal route in the graph of G_s from v_s to v_t as global path p_g. After the global path p_g from v_s to v_t is determined, the candidate subgraphs that the p_g passes through are assigned to the corresponding candidate UAVs. Then, the DAN algorithm performs continuous local path optimization in each candidate UAV.

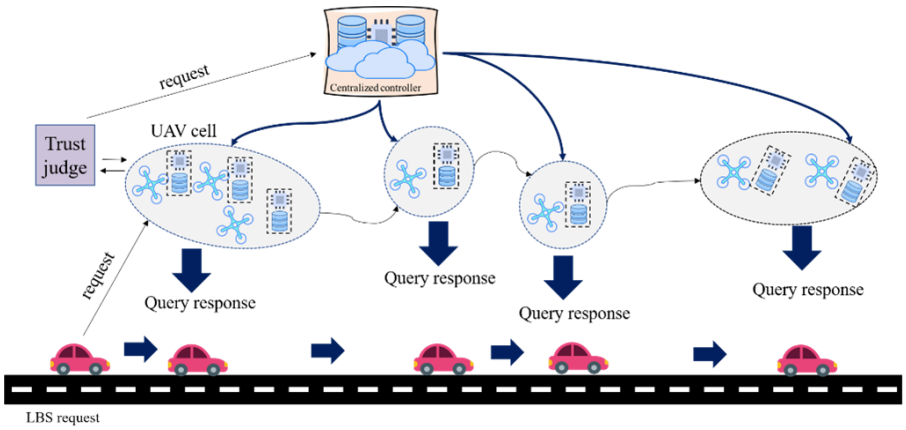


Fig. 4. Distributed framework of DAN.

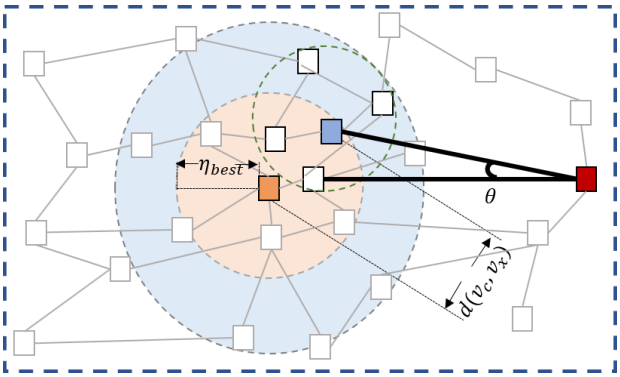


Fig. 5. Local path optimization.

It is assumed that U = U_1,U_2, . . . , U_n is the set of candidate UAVs, and local path optimization is carried out successively in the UAV $U_i (i \in [1, n])$. And we define *slot* as an intermediate node. The algorithm process is described as follows.

Algorithm 2 Local Route Optimization

Require: $q(v_s, v_t)$, $\mathcal{U}$={U_1, ⋯, U_n};
Ensure: the optimized planned route
1: **for all** U_r: $\mathcal{U}$ **do**
2: **if** v_j is covered U_r **then**
3: Set B = Breadth first traverse from v_j within 2·η_{best}
4: **for all** vertex $v_j \in$ set B **do**
5: computes $w(v_i)$
6: **end for**
7: $w(slot) = \max_{v_i \in B} \{\alpha_d \cdot w_d(v_x) + \alpha_p \cdot w_p(v_x) + \alpha_s \cdot w_s(v_x)\}$
8: compute shortest path with A* from v_c to *slot* until arrives at v_{out}
9: **end if**
10: **end for**

First, we believe that the direction from the current vertex to *slot* has a large evaluation impact. v_x is the candidate vertex when choosing a suitable *slot*. Note that, except for *start-grid* GD_s and *end-grid* GD_d, the local routing optimization in the grid starts from the border vertices and exits from the border vertices, then we call them v_{in} and v_{out} respectively. Suppose $\theta(v_s, v_t)$, (v_x, v_t) is the angle between two lines (v_s, v_t) and (v_x, v_t). The angle represents the difference between the direction pointing to *slot* and the direction pointing to v_{out}. Therefore, we prefer small angles, which means less difference in orientation. We use $w_d(v_x)$ to measure the weight of the direction as:

$$w_d(v_x) = Cos(\theta((v_{in}, v_{out}), (v_x, v_{out}))), \theta \in [0, \frac{\pi}{2}] \quad (1)$$

At the same time, the position of *slot* is also an important criterion, and we think we must choose an appropriate step size in the search. If the search step is too large, i.e. *slot* is far from the user's current location, it may cause too many recalculations due to frequently changing traffic conditions. Although the step size is very small, it may affect the performance of the local optimization because it is difficult to satisfy the global optimization in the mesh. Then we use $w_p(v_x)$ to balance computational overhead and routing performance:

$$w_p(v_x) = 1 - \frac{|\eta_{best} - d(v_c, v_x)|}{\eta_{best}}, \quad (2)$$

where η_{best} means the proper steps in search, $d(v_c, v_x)$ means the "as-the-crow-flies" distance from current position v_c to candidate vertex v_x.

Another key criterion is to choose a *slot* with more out-degrees and less weight increase in adjacent out-edges. Because a larger out-degree value means that the user has more choices when reaching *slot*, and the less weight increase of adjacent out-edges means less congestion around *slot*. Note that the value of out-degree is meaningless if all out-edges are "congested", means their weights become larger. Assuming that $V_u(v_x)$ is the set of adjacent vertices v_x. Satisfy that any $v_i \in V_u(v_x)$, $\frac{w(v_x, v_i)'}{w(v_x, v_i)} \geq \epsilon$ (we set $\epsilon = 0.8$), where $w(v_x, v_i)'$ and $w(v_x, v_i)$ represent the original and updated weights of the edge (v_x, v_i), respectively. Then the selection weight equation is:

$$w_s(v_x) = |V_u(v_x)| \cdot \max_{v_i \in V_u(v_x)} \frac{w'(v_x, v_i)}{w(v_x, v_i)} \quad (3)$$

As thus, the criteria for selecting *slots* are as follows:

$$w(v_x) = \alpha_d \cdot w_d(v_x) + \alpha_p \cdot w_p(v_x) + \alpha_s \cdot w_s(v_x) \quad (4)$$

where $w_d(v_x)$, $w_p(v_x)$ and $w_s(v_x)$ are direction weight, position weight, and selectivity weight respectively, and α_d, α_p, and α_s are user-specified parameters. In practice, we think that the three criteria are equally important for selecting a *slot*, so we set α_p, α_d, and α_s to be 1/3.

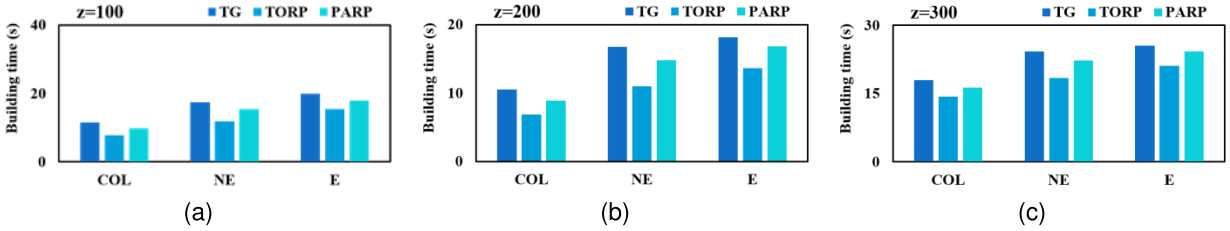


Fig. 6. Building time.

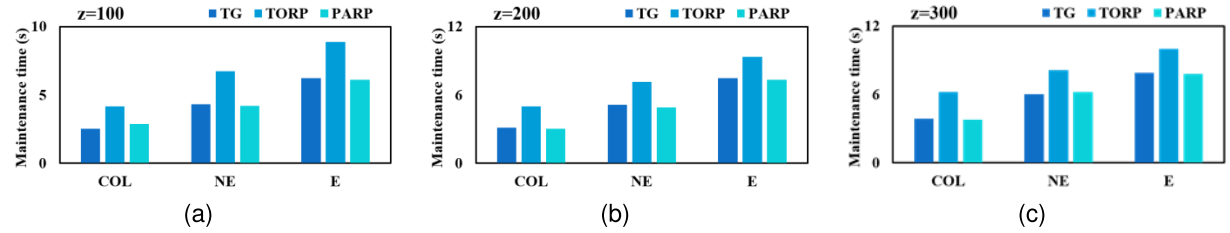


Fig. 7. Maintenance time.

TABLE I
ROAD NETWORKS DATASETS

Name	Region	#vertices	#edges
COL	Colorado	435,666	1,057,066
NE	Northeast USA	1,524,453	3,897,636
E	Eastern USA	3,598,623	8,778,114

VI. EXPERIMENTS

A. Experiment Set

The multi-threading Master-Worker model is used to realize the algorithm, the computing platform is equipped with 2.00 GHz with 4-core Intel Core, Windows 10, 16 Gigabytes. We use three road networks in Colorado (COL)[1], Northeast USA (NE)[1], and Eastern USA (E)[1] as the dataset, and the detailed information of these graphs are listed in Table I. There are two vertices included in every element of dataset and the distance between the two vertices, and we construct a directed graph using all the tuples provided in the dataset. The weights of paths are changed according to the work [48]. The results show that nearly 35% edges process the changing time of travel (simulated cost), with an average speed of 32.9 km/h and a standard deviation of 11 km/h, according to Berlin Data. Based on this rule, this paper changes the weight of 35% of the edges and sets the change rate to plus or minus 30%.

Next, we summarize the parameters that will be evaluated in the experiments.

- z represents size of subgraph.
- η_{best} represents the best search distance in local path optimization.
- N_q represents the number of queries.
- Improvement ratio λ is defined as Equation 5, which represents the optimization degree of the planned route, where p_s and p_t represent the standard path computed by Dijkstra algorithm and the planned path computed by our proposed algorithm, respectively. $TC(p_s)$ and $TC(p_t)$ separately represent the total travel cost of p_s and p_t.

$$\lambda = \frac{TC(p_s)}{TC(p_t)} \tag{5}$$

B. Baselines

The baseline methods are as follows:

- **Naive** A rough force way is used by naive method to find out the route optimization in dynamic road network.
- **TORP** Song et al. [46] put forward a new-style parallel path planning method that can identify the vicious nodes and the time cost in path planning is very low. In TORP, they introduce DTP index over dynamic road network to decompose complex tasks into multiple lightweight subtasks which is quite efficient.
- **PARP** Dai et al. [41] take the traffic conditions information of past, present, and future into count by introducing dual-level path index DLP and GCN model to find out the path optimization in parallel constantly.

C. Evaluation of TG-Index

1) Building Time: Firstly, we compare the building time of TG-index, TORP, and PARP under different subgraph sizes z, as shown in Fig. 6. It shows that the building time of the TG-index grows with the extent of the graph synchronously, that is, for larger-scale graphs, more sub-graphs must be constructed, and more k shortest paths are calculated at the same time. As thus, it consumes more time. Meanwhile, the building time of TG-index is little more than that of TORP and PARP, because TG-index requires a prefix path tree for the storage of K shortest paths, while TORP only calculates the k shortest paths and PARP calculates the top-k frequent paths. Once the index is constructed, the maintenance time of TG-index is much smaller than that of TORP, and the experimental results VI-C.2 can testify that conclusion.

2) Maintenance Time: The effects of different subgraph sizes z on index maintenance are evaluated and we compare with TORP and PARP method. The results can be seen in Fig. 7. It can be shown that the amount of maintenance time of

[1]http://users.diag.uniroma1.it/challenge9

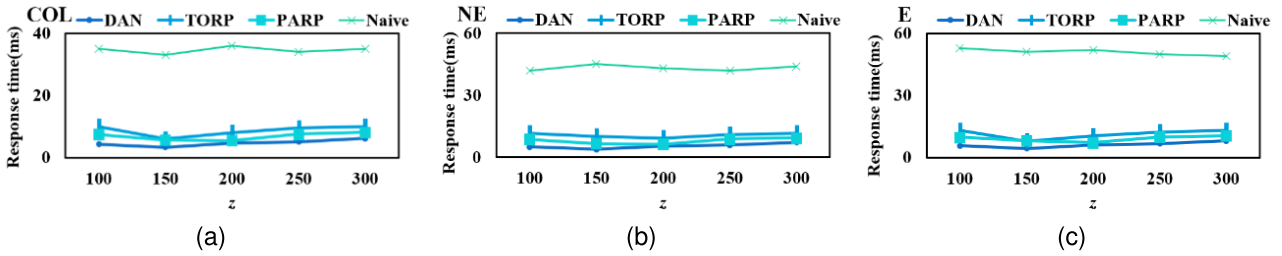


Fig. 8. Response time.

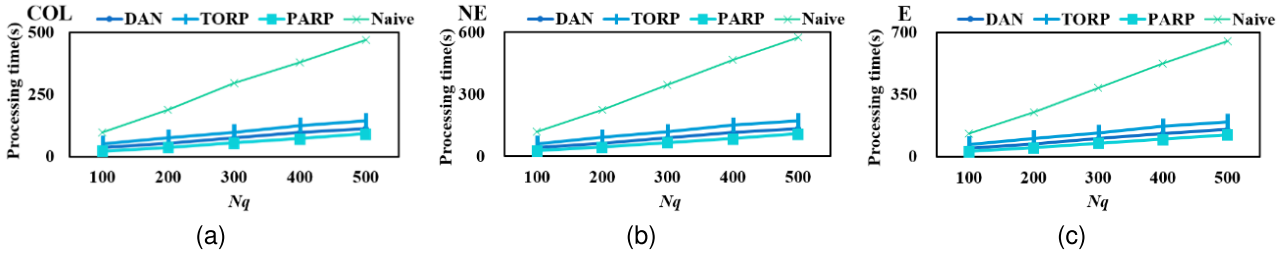


Fig. 9. Processing time.

TABLE II
λ FOR VARYING η_{best}

η_{best}	50	100	150	200	250
$\lambda(COL)$	1.213	1.422	1.463	**1.521**	1.413
$\lambda(NE)$	1.422	**1.552**	1.412	1.324	1.302
$\lambda(E)$	1.243	1.340	1.424	**1.632**	1.528

TORP is almost 40%-50% more than that of TG-index. This is because when the road conditions keep changing, TORP must frequently calculate k shortest paths between border vertices, which cost heavy maintenance time. Meanwhile, the maintenance time of PARP and TG-index are almost equivalent, because they both use prefix tree path to store and update k paths.

D. Evaluation of DAN

1) Effect of η_{best}: Table II depicts the influence of parameters η_{best} on the improvement ratio λ. A bigger value of λ means the better performance of our proposal in terms of reducing the travel cost of the planned path. With the value of η_{best} increasing, the λ on each road network generally grows firstly and then begins to decrease slowly. The results of evaluation show that the values of η_{best} cannot be too large or too small. It is clear that the greatest values of λ on COL, NE, and E are 1.521, 1.552, and 1.632, respectively when η_{best} takes the proper values. Hereinafter, η_{best} takes values 200, 100, 200 on COL, NE, and E separately unless otherwise specified.

2) Response Time: The response time of the route planning query refers to the average time consumed by each optimization. Corresponding to different subgraph sizes, the response time of DAN, TORP, PARP and Naive on each graph is tested, and the results are shown in Fig. 8. First, we feed a batch of query (Nq = 100), and then measure the average response time of these queries. According to the results, it can be seen that the response time of DAN is smaller than that of TORP and PARP, and moreover, when the subgraph size increases, the response time of DAN decreases slightly at first, and then gradually increases. This is because when the subgraph is too small, more local path optimizations are done in the route planning query. In this case, the cost of scheduling multiple threads also increases, as does the average response time for each optimization. As the z increases, the number of local optimizations for the query will decrease, and the thread allocation cost will also decrease, resulting in faster response time. However, when the subgraph size grows above a certain threshold (e.g., 150 on COL), the computation of the shortest path in each local optimization incurs more cost and dominates the total cost. Therefore, as the subgraph gets larger, the response time starts to increase. Since Naive does not involve partitioning, the subgraph size has no influence on the response time.

3) Processing Time: Then we test the processing time of these four methods. The processing time refers to the sum of the time consumed in processing the query. Firstly, we send a batch of queries (Nq = 100), and then measure the average processing time of these queries. As shown in Fig. 9, it can be seen that the processing time of each method increases with the number of queries, and the processing time of Naive is significantly greater than that of DAN, TORP and PARP. At the same time, the processing time of DAN is between TORP and PARP, and the processing time of these three methods is similar.

4) Travel Cost: Fig. 10 shows the influence of the path length on the travel cost. The principle of DAN is to continuously optimize the planned route on the dynamic graph with a shorter response time. Please note that it is impossible to identify the absolute best (shortest) path on the dynamic graph, because the future traffic condition cannot be accurately predicted. It can be seen that the travel cost increases significantly with the increase of the path length. Although TORP and Naive always recalculate shortest paths, and DAN, TORP, and PARP all optimize the planned path in local area, the travel cost of these four methods is almost equivalent. This

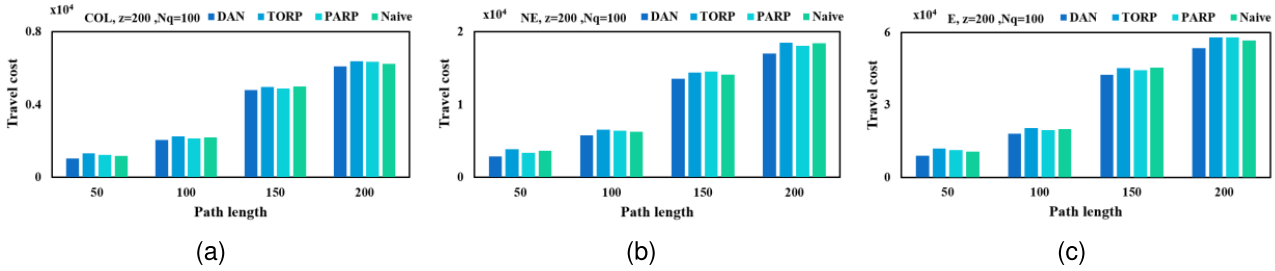


Fig. 10. Travel time.

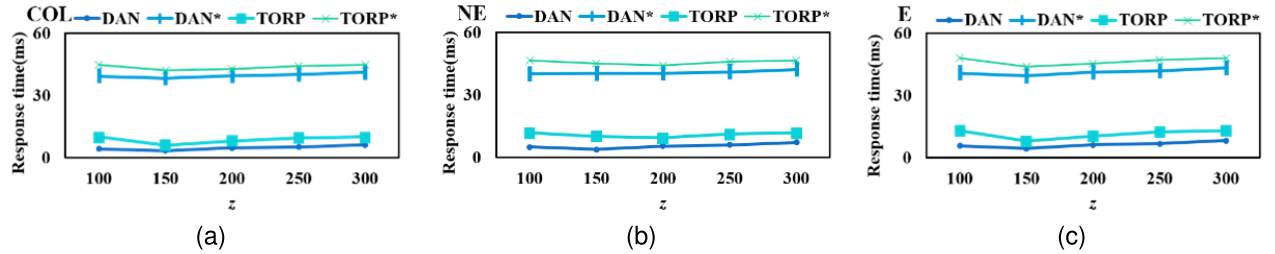


Fig. 11. Response time w.r.t trust model.

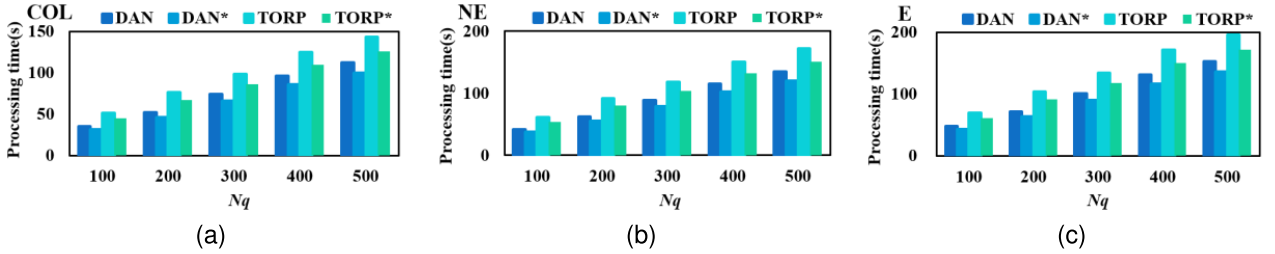


Fig. 12. Processing time w.r.t trust model.

is because DAN, TORP and PARP optimize planned paths locally, which can better bypass congested roads in advance.

E. Evaluation of Trust Model

In this section, we will introduce the performance of Trust Model enabled route planning algorithm. We evaluate our algorithm under two types of attack mode, namely *attack mode_a* and *attack mode_b*.

- *Attack mode_a* is a relatively simple mode of attack, which is to show malicious behaviors all the time. A vehicle in mode_a sends query requests with high frequency, or in unreasonable places, such as in grass or buildings.
- *Attack mode_b* will alternately show malicious behaviors and honest behaviors. In detail, a vehicle in mode_b intermittently performs mode_a attacks to avoid detection.

We evaluate the response time and processing time of the system with Trust Model under two types of attack mode, namely *attack mode_a* and *attack mode_b*, and the results are shown in Fig. 11 and Fig. 12. We can observe that the methods with Trust Model cost more response time for each optimization, as the process of trust judgment causes extra time cost. When the subgraph size increases, the response time decreases slightly at first, and then gradually increases. This is because when the subgraph is too small, more local path optimizations are done in the route planning query. In this case, the cost of scheduling multiple threads also increases, as does the average response time for each optimization. As the subgraph size increases, the number of local optimizations for the query will decrease, and the thread allocation cost will also decrease, resulting in faster response time. Meanwhile, we can find that the processing time of Trust Model-enabled methods is significantly lower than that of methods with non-Trust Model. We can also find that the processing time of each method increases with the number of queries, the processing time of Trust Model enabled method is significantly greater, and DAN performs less processing time than that of TORP. This is because Trust Model can effectively detect malicious vehicles which reduces the waste of computing resources.

VII. Conclusion

The transport environment of the increasing unmanned devices is full of uncertainty, ambiguity, and vague information. The problem of route planning over AGIN on dynamic road networks is the basis for solving the decision of intelligent transportation. However, the challenges of processing a large number of concurrent queries are ignored by most existing works. Given these issues, our work proposes TG-index that provides cascade pruning capabilities to improve the efficiency of route planning. Meanwhile, we introduce Trust Model to efficiently find malicious vehicles and save computing resources. On the basis of TG-index and Trust Model, a DAN model is proposed to continuously optimize paths in parallel on the dynamic graph. Finally, an extensive evaluation is

conducted to confirm the effectiveness and superiority of our proposal.

As future work, we will consider changes in future road traffic conditions combined with deep learning model. In practice, the problem of shared computing for multiple concurrent queries can be considered. It is worthwhile to study a solution to consider future road traffic conditions and optimize the solution of shared paths for multiple concurrent queries.

REFERENCES

[1] X. Li, X. Ma, and X. Wang, "A survey of path planning algorithms for mobile robots," *Comput. Meas. Control*, vol. 30, no. 7, pp. 1–11, 2022.

[2] S. Shao, W. Guan, B. Ran, Z. He, and J. Bi, "Electric vehicle routing problem with charging time and variable travel time," *Math. Problems Eng.*, vol. 2017, pp. 1–13, Jan. 2017.

[3] J. Xu, L. Guo, Z. Ding, X. Sun, and C. Liu, "Traffic aware route planning in dynamic road networks," in *Proc. Int. Conf. Database Syst. Adv. Appl.*, 2012, pp. 576–591.

[4] T. Liebig, N. Piatkowski, C. Bockermann, and K. Morik, "Dynamic route planning with real-time traffic predictions," *Inf. Syst.*, vol. 64, pp. 258–265, Mar. 2017.

[5] A.-E.-M. Taha, "Facilitating safe vehicle routing in smart cities," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–5.

[6] D. Tischner, "Multi-modal route planning in road and transit networks," 2018, *arxiv:1809.05481*.

[7] A. Özal, A. Ranganathan, and N. Tatbul, "Real-time route planning with stream processing systems: A case study for the city of lucerne," in *Proc. 2nd ACM SIGSPATIAL Int. Workshop GeoStreaming (IWGS)*, 2011, pp. 21–28.

[8] U. Demiryurek, F. Banaei-Kashani, C. Shahabi, and A. Ranganathan, "Online computation of fastest path in time-dependent spatial networks," in *Proc. Int. Symp. Spatial Temporal Databases*. Berlin, Germany: Springer, 2011, pp. 92–111.

[9] H. Wang et al., "R3: A real-time route recommendation system," *Proc. VLDB Endowment*, vol. 7, no. 13, pp. 1549–1552, Aug. 2014.

[10] N. Malviya, S. Madden, and A. Bhattacharya, "A continuous query system for dynamic route planning," in *Proc. IEEE 27th Int. Conf. Data Eng.*, Apr. 2011, pp. 792–803.

[11] D. Zhang, D. Yang, Y. Wang, K.-L. Tan, J. Cao, and H. T. Shen, "Distributed shortest path query processing on dynamic road networks," *VLDB J.*, vol. 26, no. 3, pp. 399–419, Jun. 2017.

[12] Y. Tong, Y. Zeng, Z. Zhou, L. Chen, J. Ye, and K. Xu, "A unified approach to route planning for shared mobility," *Proc. VLDB Endowment*, vol. 11, no. 11, pp. 1633–1646, 2018.

[13] M. Holzer, F. Schulz, and D. Wagner, "Engineering multilevel overlay graphs for shortest-path queries," *ACM J. Experim. Algorithmics*, vol. 13, pp. 156–170, Feb. 2009.

[14] H. Zhu, W. Li, W. Liu, J. Yin, and J. Xu, "Top K optimal sequenced route query with POI preferences," *Data Sci. Eng.*, vol. 7, no. 1, pp. 3–15, Mar. 2022.

[15] Y. Wang, Y. Yuan, Y. Ma, and G. Wang, "Time-dependent graphs: Definitions, applications, and algorithms," *Data Sci. Eng.*, vol. 4, no. 4, pp. 352–366, Dec. 2019.

[16] S. Wang, J. Cao, and P. S. Yu, "Deep learning for spatio-temporal data mining: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 8, pp. 3681–3700, Aug. 2022.

[17] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Autonomous Robot Vehicles*. New York, NY, USA: Springer, 1986, pp. 396–404.

[18] G. Li, A. Yamashita, H. Asama, and Y. Tamura, "An efficient improved artificial potential field based regression search method for robot path planning," in *Proc. IEEE Int. Conf. Mechatronics Autom.*, Aug. 2012, pp. 1227–1232.

[19] Z. Liu, X. Yuan, G. Huang, Y. Wang, and X. Zhang, "Two potential fields fused adaptive path planning system for autonomous vehicle under different velocities," *ISA Trans.*, vol. 112, pp. 176–185, Jun. 2021.

[20] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numer. Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959.

[21] C. Zhang, L. Xin, and Z. Xiangwei, "Electricity transmission line path planning based on improved Dijkstra algorithm," *Electric Power Surv. Design*, vol. 164, no. 2, pp. 1–5, 2022.

[22] Y. Chen, "Application of improved Dijkstra algorithm in coastal tourism route planning," *J. Coastal Res.*, vol. 106, no. SI, pp. 251–254, 2020.

[23] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-4, no. 2, pp. 100–107, Jul. 1968.

[24] H. Min, X. Xiong, P. Wang, and Y. Yu, "Autonomous driving path planning algorithm based on improved A* algorithm in unstructured environment," *Proc. Inst. Mech. Engineers, D, J. Automobile Eng.*, vol. 235, nos. 2–3, pp. 513–526, Feb. 2021.

[25] A. Sharma and N. Kumar, "Third eye: An intelligent and secure route planning scheme for critical services provisions in Internet of Vehicles environment," *IEEE Syst. J.*, vol. 16, no. 1, pp. 1217–1227, Mar. 2022.

[26] W. Dong, M. Wozniak, J. Wu, W. Li, and Z. Bai, "De-noising aggregation of graph neural networks by using principal component analysis," *IEEE Trans. Ind. Informat.*, early access, Mar. 7, 2022, doi: 10.1109/TII.2022.3156658.

[27] Mallika, J. S. Ubhi, and A. K. Aggarwal, "Neural style transfer for image within images and conditional GANs for destylization," *J. Vis. Commun. Image Represent.*, vol. 85, May 2022, Art. no. 103483.

[28] A. Colorni, M. Dorigo, and V. Maniezzo, "Distributed optimization by ant colonies," in *Proc. 1st Eur. Conf. Artif. Life*, vol. 142, Dec. 1991, pp. 134–142.

[29] S. Viswanathan, K. S. Ravichandran, A. M. Tapas, and S. Shekhar, "An intelligent gain based ant colony optimisation method for path planning of unmanned ground vehicles," *Defence Sci. J.*, vol. 69, no. 2, pp. 167–172, Mar. 2019.

[30] X. Dai, S. Long, Z. Zhang, and D. Gong, "Mobile robot path planning based on ant colony algorithm with A* heuristic method," *Frontiers Neurorobotics*, vol. 13, p. 15, Apr. 2019.

[31] Y. Guan, M. Gao, and Y. Bai, "Double-ant colony based UAV path planning algorithm," in *Proc. 11th Int. Conf. Mach. Learn. Comput. (ICMLC)*, 2019, pp. 258–262.

[32] H. J. Bremermann, "The evolution of intelligence: The nervous system as a model of its environment," Dept. Math., Univ. Washington, Seattle, WA, USA, Tech. Rep. 1, NBS 5800491, 1958.

[33] H. Qu, K. Xing, and T. Alexander, "An improved genetic algorithm with co-evolutionary strategy for global path planning of multiple mobile robots," *Neurocomputing*, vol. 120, pp. 509–517, Nov. 2013.

[34] M. Nazarahari, E. Khanmirza, and S. Doostie, "Multi-objective multi-robot path planning in continuous environment using an enhanced genetic algorithm," *Exp. Syst. Appl.*, vol. 115, pp. 106–120, Jan. 2019.

[35] L.-N. Gao et al., "A short-distance healthy route planning approach," *J. Transp. Health*, vol. 24, Mar. 2022, Art. no. 101314.

[36] M. Woźniak, A. Zielonka, and A. Sikora, "Driving support by type-2 fuzzy logic control model," *Expert Syst. Appl.*, vol. 207, Nov. 2022, Art. no. 117798.

[37] A. Kumar, Y. Sato, T. Oishi, S. Ono, and K. Ikeuchi, "Improving GPS position accuracy by identification of reflected GPS signals using range data for modeling of urban structures," *SEISAN KENKYU*, vol. 66, no. 2, pp. 101–107, 2014.

[38] S. Wu, Y. Li, H. Zhu, J. Zhao, and G. Chen, "Dynamic index construction with deep reinforcement learning," *Data Sci. Eng.*, vol. 7, no. 2, pp. 87–101, Jun. 2022.

[39] A. Mondal, A. Kakkar, N. Padhariya, and M. Mohania, "Efficient indexing of top-K entities in systems of engagement with extensions for geo-tagged entities," *Data Sci. Eng.*, vol. 6, no. 4, pp. 411–433, Dec. 2021.

[40] T. Dai et al., "Continuous route planning over a dynamic graph in real-time," *Proc. Comput. Sci.*, vol. 174, pp. 111–114, Jan. 2020.

[41] T. Dai, B. Li, Z. Yu, X. Tong, M. Chen, and G. Chen, "PARP: A parallel traffic condition driven route planning model on dynamic road networks," *ACM Trans. Intell. Syst. Technol.*, vol. 12, no. 6, pp. 1–24, Dec. 2021.

[42] S. Wang, J. Cao, H. Chen, H. Peng, and Z. Huang, "SeqST-GAN: Seq2Seq generative adversarial nets for multi-step urban crowd flow prediction," *ACM Trans. Spatial Algorithms Syst.*, vol. 6, no. 4, pp. 1–24, 2020.

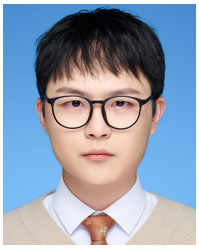
[43] M. Rezaei, H. Noori, M. Mohammadkhani Razlighi, and M. Nickray, "ReFOCUS+: Multi-layers real-time intelligent route guidance system with congestion detection and avoidance," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 1, pp. 50–63, Jan. 2021.

[44] J. Xie et al., "A survey on machine learning-based mobile big data analysis: Challenges and applications," *Wireless Commun. Mobile Comput.*, vol. 2018, Aug. 2018, Art. no. 8738613.

[45] J. Li et al., "An end-to-end load balancer based on deep learning for vehicular network traffic control," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 953–966, Feb. 2019.

[46] X. Song, B. Li, T. Dai, and J. Tian, "A trust management-based route planning scheme in LBS network," in *Proc. Int. Conf. Adv. Data Mining Appl.* Cham, Switzerland: Springer, 2022, pp. 307–322.
[47] J. Y. Yen, "An algorithm for finding shortest routes from all source nodes to a given destination in general networks," *Quart. Appl. Math.*, vol. 27, no. 4, pp. 526–530, 1970.
[48] Z. Yu et al., "Distributed processing of K shortest path queries over dynamic road networks," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Jun. 2020, pp. 665–679.

Ken Cai received the Ph.D. degree in biomedical engineering from the South China University of Technology, China, in 2011. He is currently a Professor with the College of Automation, Zhongkai University of Agriculture and Engineering, China. His current main research interests include intelligent transportation, medical data analytic, deep learning, pattern recognition, decision making, and biomedical instrument.



Tianlun Dai received the B.S. degree in computer science and technology from the University of Jinan, Jinan, China, in 2020. He is currently pursuing the master's degree with the Nanjing University of Aeronautics and Astronautics, Nanjing, China. His research interests include spatio-temporal database and automatic navigation. He is a Student Member of CCF.

Qinyong Lin received the B.S. and Ph.D. degrees in biomedical engineering from the South China University of Technology, Guangzhou, China, in 2010 and 2016, respectively. He was a Post-Doctoral Research Associate with the South China University of Technology from 2017 to 2019. He is currently an Associate Professor with the College of Automation, Zhongkai University of Agriculture and Engineering, Guangzhou. His main research interests include computer vision, artificial intelligence, the Internet of Things, data mining, surgical robot, and image-guided navigation.

Xinyang Song received the B.S. degree in computer science and technology from the Shandong University of Finance and Economics, Jinan, China, in 2020. He is currently pursuing the master's degree with the Nanjing University of Aeronautics and Astronautics, Nanjing, China. His research interests include privacy protection of blockchain. He is a Student Member of CCF.

Wei Zhou received the master's degree from the PLA University of Science and Technology, Nanjing, China, in 2004. He is currently pursuing the Ph.D. degree with the Nanjing University of Aeronautics and Astronautics. His research interests include software theory, data security, and big data.

Qian Zhou received the Ph.D. degree in computer application from the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics (NUAA), China, in 2018. She is currently a Lecturer with the School of Modern Posts and the Institute of Modern Posts, Nanjing University of Posts and Telecommunications (NJUPT). Her current main research interests include cryptography, network security and privacy, the Internet of Things Technology, database, and other basic theories and applications.

Jinzhan Wei received the B.S. degree in surveying engineering and the M.S. degree in geodesy and survey engineering from Central South University, Changsha, China, in 2002 and 2005, respectively. He is currently a Professor with the School of Electronic Information and Automation, Guilin University of Aerospace Technology, Guilin, China. His main research interests include spatial information science.

Feng Wang received the master's degree in software engineering from Hohai University (HHU), Nanjing. He is currently pursuing the Ph.D. degree in computer application with the Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing. His current research interests include spatiotemporal database, blockchain, and knowledge graph.

Huazhou Chen received the B.S. degree in thermal energy and power engineering from Nanchang University, Nanchang, China, in 2006, the M.S. degree in mathematics from Jinan University, Guangzhou, China, in 2008, and the Ph.D. degree in applied mathematics from Shanghai University, Shanghai, China, in 2011. He was a Visiting Scholar with the Department of Statistical Science, University College London, U.K., from 2015 to 2016. He is currently a Professor with the College of Science, Guilin University of Technology, Guilin, China. His main research interests include computational biology, machine learning, pattern recognition, chemominformatics, and spectrometry.

Bohan Li received the Ph.D. degree in computer application from the Harbin University of Science and Technology, Harbin, in 2009. He is currently an Associate Professor at the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing. His current research interests include spatio-temporal database, the Internet of Things, and knowledge graph. He is a member of CCF and ACM.